AFRL-IF-RS-TR-1998-183
Final Technical Report
September 1998

SECURE HETEROGENEOUS APPLICATION RUN-TIME ENVIRONMENT (SHARE)

Sanders, A Lockheed Martin Company

Sponsored by
Defense Advanced Research Projects Agency
DARPA Order No. D355

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the U.S. Government.

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

AFRL-IF-RS-TR-1998-183 has been reviewed and is approved for publication.

APPROVED:

RALPH KOHLER
Project Engineer

FOR THE DIRECTOR:

NORTHRUP FOWLER, III, Technical Advisor
Information Technology Division
Information Directorate

If your address has changed or if you wish to be removed from the Air Force Research Laboratory Rome Research Site mailing list, or if the addressee is no longer employed by your organization, please notify AFRL/IFTC, 26 Electronic Pky, Rome, NY 13441-4514. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document require that it be returned.

SECURE HETEROGENEOUS APPLICATION RUN-TIME ENVIRONMENT (SHARE)

Jeff Smith

Contractor:	Sanders, A Lockheed Martin Company
Contract Number:	F30602-95-2-0051
Effective Date of Contract:	29 August 1995
Contract Expiration Date:	11 November 1997
Short Title of Work:	Secure Heterogeneous Application Run-Time Environment (SHARE)
Period of Work Covered:	Aug 95 – Nov 97

Principal Investigator:	Jeff Smith
Phone:	(603) 885-3505
AFRL Project Engineer:	Ralph Kohler
Phone:	(315) 330-2016

Approved for public release; distribution unlimited

This research was supported by the Defense Advanced Research Projects Agency of the Department of Defense and was monitored by Ralph Kohler, AFRL/IFTC, 26 Electronic Pky, Rome, NY.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	September 1998	Final Aug 95 - Nov 97

4. TITLE AND SUBTITLE

SECURE HETEROGENEOUS APPLICATION RUN-TIME ENVIRONMENT (SHARE)

5. FUNDING NUMBERS

C - F30602-95-2-0051
PE - 62301E
PR - D002
TA - 02
WU - P1

6. AUTHOR(S)

Jeff Smith

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Sanders, A Lockheed Martin Company
Signal Processing Center, PTP02-B002
Advanced Engineer & Technology division
P.O. Box 868
Nashua NH 03061-0868

8. PERFORMING ORGANIZATION REPORT NUMBER

N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Defense Advanced Research Projects Agency
3701 North Fairfax Drive
Arlington VA 22203-1714

AFRL/IFTC
26 Electronic Pky
Rome NY 13441-4514

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

AFRL-IF-RS-TR-1998-183

11. SUPPLEMENTARY NOTES

AFRL Project Engineer: Ralph Kohler/IFTC/(315) 330-2016

12a. DISTRIBUTION AVAILABILITY STATEMENT

Approved for Public Release; Distribution Unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The SHARE (Secure Heterogeneous Application Run-Time Environment for High-Performance Scalable Computing) is a contractual cooperative agreement between Sanders and DARPA. The SHARE program addresses the critical requirements that will enable next-generation Command, Control, Communications and Intelligence (C3I) and combat systems to incorporate low-cost distributed High Performance Scalable Computing (HPSC) technology. Research efforts with the high-performance PacketWay protocol, Myricom packet-switching network technology, and the Message Passing Interface (MPI) standard have been combined, adapted, and extended to create a secure, inter-operable heterogeneous application run-time environment.

14. SUBJECT TERMS

SHARE tasks, PacketWay, Alternative Architectures, SHARE SAN Communication, Key Management Plan Development, SHARE*HPSC Network Security Architecture

15. NUMBER OF PAGES

236

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

Table of Contents

Secure Heterogeneous Application Run-Time Environment (SHARE) Final Report

Abstract

The SHARE (Secure Heterogeneous Application Run-time Environment For High-Performance Scalable Computing) is a contractual cooperative agreement between Sanders and DARPA, which started on 29 August 1995 and ended the end of November 1997. The SHARE program addresses the critical requirements that will enable next-generation Command, Control, Communications and Intelligence (C^3I) and combat systems to incorporate low-cost distributed High Performance Scalable Computing (HPSC) technology. Research efforts with the high-performance PacketWay protocol, Myricom packet-switching network technology, and the Message Passing Interface (MPI) standard have been combined, adapted, and extended to create a secure, inter-operable heterogeneous application run-time environment (see Figure 1). There are three major SHARE tasks.

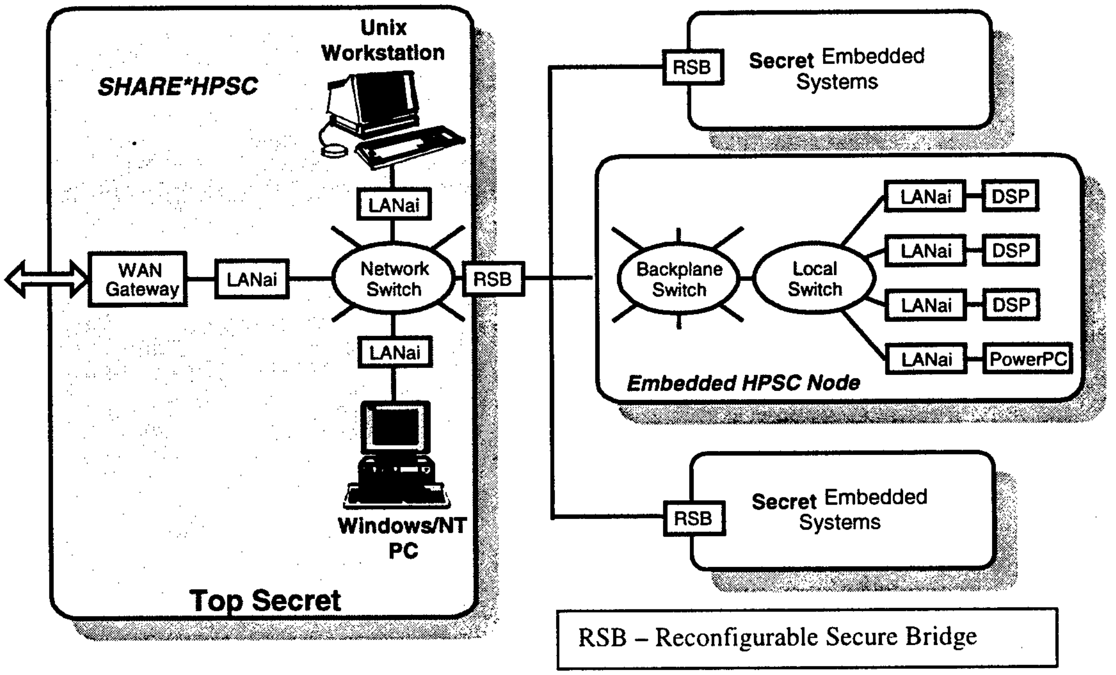


Figure 1 SHARE Environment

1. Develop a portable run-time software environment including an extended message-passing library, device drivers, and host interface logic for embedded HPSC systems.

2. Perform PacketWay/MPI standards adaptations to support data security, traffic latency minimization, real-time pre-emption, and optimal allocation of computational processing and memory resources for distributed combat systems.

3. Demonstrate the run-time environment on a heterogeneous HPSC prototype network.

The SHARE run-time environment combines PacketWay and MPI technology to provide an integrated run-time environment that encapsulates and abstracts the HPSC hardware module-level details without an implied loss of performance. Multi-layered control software manages node-to-node, system-to-node, and system-level issues related to heterogeneous node inter-operability, security, latency, and efficient utilization/access of computational resources. In addition, the use of MPI as a programmatic paradigm for applications software provides long-term maintainability and portability of application libraries. Sanders ongoing HPSC prototype development has been integrated with a real-time heterogeneous local area network to demonstrate the SHARE concept.

1 Need

DARPA is developing enabling technology for interconnecting HPSC systems via networking schemes offering substantially improved performance over currently available technologies. Commercial and defense applications of these HPSC systems require a secure integrated run-time software environment. This contract has focused on the development of this secure run-time environment and its transition to commercial and defense applications.

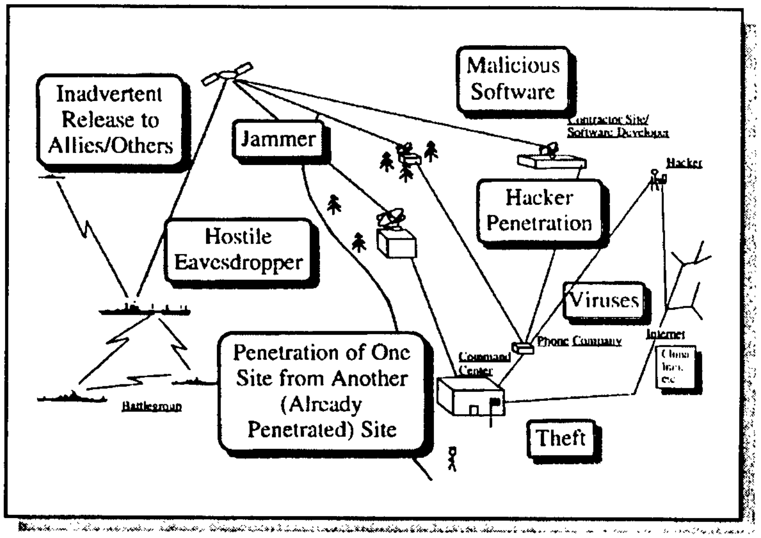

Inadvertent Release to Allies/Others
Jammer
Malicious Software
Contractor Site/ Software Developer
Hacker
Hacker Penetration
Hostile Eavesdropper
Viruses
Phone Company
Internet
Command Center
Penetration of One Site from Another (Already Penetrated) Site
Battlegroup
Theft

Figure 2 Problems with Today's Systems

2 Objective

The overall objective of the SHARE research program is to develop a high bandwidth secure System Area Network (SAN) and compatible run-time environment. This run-time environments includes an extended message-passing library, device drivers, and host interface logic for the development of secure, heterogeneous embedded HPSC systems, consisting of mixed workstations and dedicated signal processors. In order to develop this secure real-time message transfer capability between SANs of multiple levels of security, our goals were to:

- Design a high speed cryptographic module derived from application specific goals,
- Propose modifications to the PacketWay and MPI standards for security, minimized traffic latency and support for real-time pre-emption,
- Demonstrate heterogeneity and security in an HPSC environment and

- Document/Publish our results

Detailed objectives of the SHARE Program were:

1. **Requirements Analysis -** Coordinate with various C^3I and combat system architectures to develop/document specific design requirements and establish a representative demonstration program based on various platform's processing/secure/message traffic needs.
2. **Network/Gateway Security Software Development -** Define and document the security requirements, perform trade-off studies to determine optimal combinations of throughput, speed, system flexibility and unit cost and develop supporting software as needed.
3. **Heterogeneous Environments -** Port libraries and develop device drivers for three operating system environments: SPARC and Pentium Solaris, and Pentium Windows NT.
4. **Demonstration -** Conduct an interim demonstration of linked HPSC processing nodes and UNIX workstations.

3 Approach

The following steps describe the approach the SHARE team followed in implementing the objectives outlined in Section 2.0.

1. Derive requirements definition and functional design from air, ship and underwater applications
2. Construct network security architecture and refine with NSA, ARPA, and RL
3. Perform high level and detailed cryptographic module design
4. Prototype and demonstrate recommended PacketWay and MPI extensions and support working validated extensions into the respective standards groups
5. Perform network simulation of a secure HPSC testbed, simulating cryptography module design, message passing schedule and network protocol

Section 4 and its subsections provide a description of the accomplishments of the SHARE program.

4 Accomplishments

The following is an outline of the major accomplishments of the SHARE program during the lifetime of the program.

- **PacketWay**
  - Documented and proposed security extensions to the PacketWay standard. These extensions have been accepted by the IETF working group.
  - Implemented two independent implementations of the PacketWay standard versions (one at Sanders, one at Mississippi State University) in support of the

IETF requirement that two implementations be in place before PacketWay can be accepted as an IETF standard.

  - Implemented secure extensions to the PacketWay standard.
  - Provided feedback on the specification to the PacketWay working group based on the use of formal methods to verify the correctness of the PacketWay specification.
  - Developed an on-line PacketWay video demonstrating the features of PacketWay.
  - Demonstrated PacketWay inter-operability between two independently developed PacketWay implementations.
- **MPI**
  - Proposed a subset of MPI functions appropriate for embedded systems which has been accepted into the MPI/RT (real-time MPI) document as MPI/RC (resource constrained MPI).
  - Implemented the MPI/RC subset on the Sanders HPSC APU (Arithmetic Processing Unit) board, Sun workstations, and Myricom LANai processor.
  - Proposed MPI security extensions to the MPI standard.
- **Secure Architecture**
  - Completed detailed design of the SHARE INFOSEC Module (SIM)
  - Completed Key Management Plan.
  - Fabricated and assembled four SIM boards
  - Defined a generic, network protocol independent, message passing interface between the SIM and the SHARE router
  - Established Memorandum Of Understanding (MOU) with NSA allowing classified Parade KG devices to be transferred to L-3 for integration and test in the SIM
  - Completed comparison study of the security features provided in Secure PacketWay, IPv6, and Space Communications Protocol Standards (SCPS)
- **Network Simulation -** Development of a library of Ptolemy simulation modules for the SHARE network.
- **Software Framework for the development of Communication Software for Two-level Multi-computers**
  - Investigated approaches to providing a framework for developing communication software on two-level multi-computers.
  - Prototyped a software framework on the HPSC APU.
- **Standards Participation**. The SHARE team has been heavily involved in various standards committees including:
  - MPI Forum
  - Real-time MPI Forum
  - IETF PacketWay working group
- **Documentation** includes the following:
  - Applications survey
  - Secure PacketWay Description
  - Network Security Architecture
  - Network Simulation Plan
  - System Requirements
  - Secure/Real-time MPI Description
  - Technology Insertion Plan
  - Commercial Cryptography Module

- Key Management Plan
- Comparison of PacketWay and IPv6

Design

- System Requirements
- Video of a SHARE Demo

4.1 PacketWay

4.1.1 What is PacketWay

PacketWay is an open family of specifications for inter-networking high-performance System Area Networks (SANs) and high-performance LANs.

PacketWay is designed to provide a uniform interface for a wide variety of SANs, such that higher level protocols (such as IP) are able to use SANs in a uniform manner. An IP driver for PacketWay would be able to use PacketWay between heterogeneous processors as if they were all on a single SAN. Figure 3 shows the relationship the PacketWay protocol and other protocols within the ISO OSI reference model.

PacketWay is designed to provide inter-operability among closely located heterogeneous processors at high speed. PacketWay sacrifices scalability and some generality for high performance. Hence, PacketWay supplements IP for high performance and for fine granularity of processors.

The PacketWay specification defines:

1. End-to-End protocol and packet format (EEP)
2. Router-to-Router protocol and packet format (RRP)
3. Resource discovery and allocation

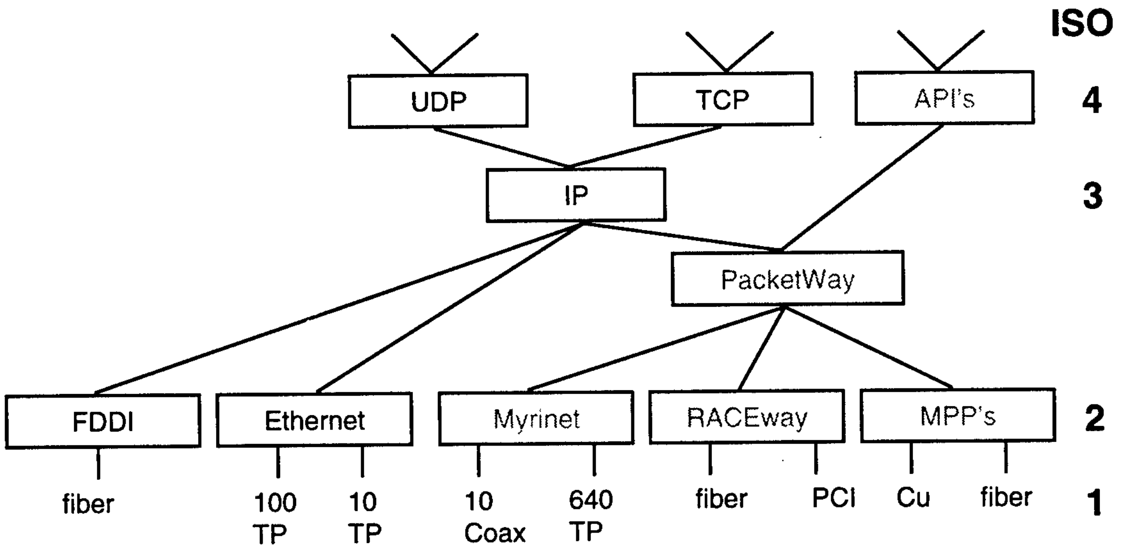


Figure 3 PacketWay's Location within the ISO OSI Reference model

4.1.2 Secure PacketWay

Secure PacketWay specifies how trusted (red) PacketWay nodes communicate over an untrusted (black) network by ensuring confidentiality, non-repeatability, and authentication (see Figure 4).

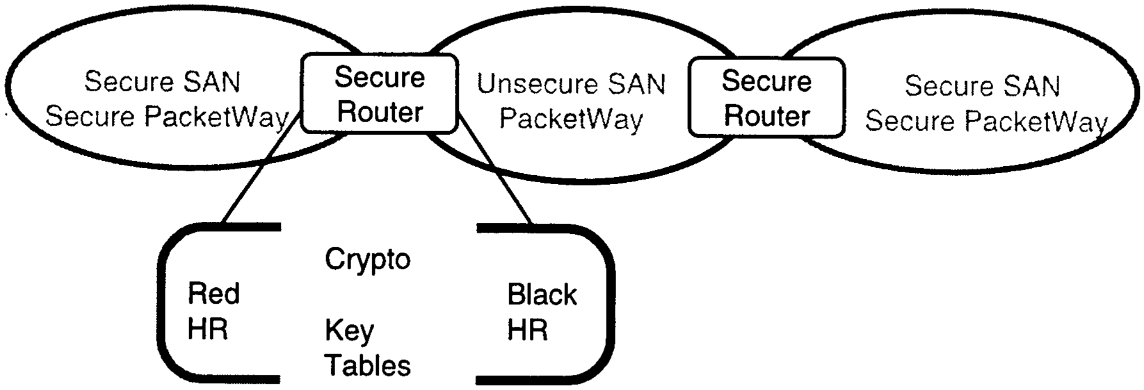


Figure 4 Secure PacketWay Network

To the end-point nodes on the red networks, the untrusted black network is a logical interconnect between two "secure half-routers". These are actually special routers that handle the details of cryptography and key management to minimize the performance overhead for the client nodes.

4.1.3 PacketWay Implementations

One of the goals of the SHARE program was to develop two interoperable PacketWay implementations in support of the IETF standardization process. The two implementations were developed independently in accordance with the IETF standardization process. An inter-operability test between these two implementations

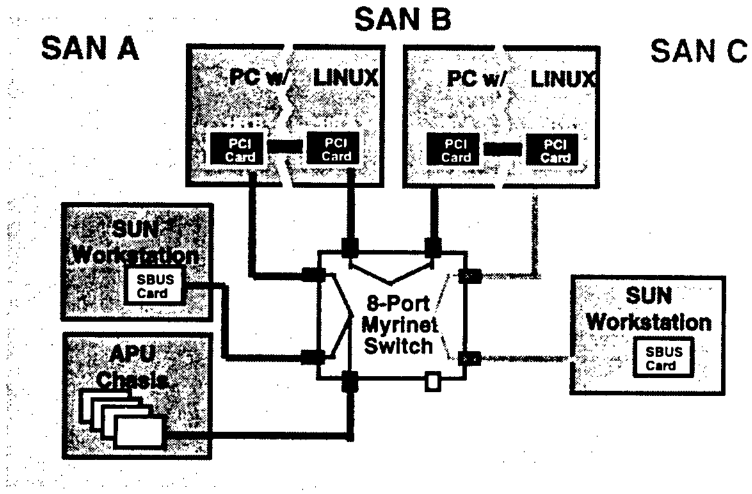


Figure 5 Sanders SHARE Laboratory Configuration

demonstrated the use of PacketWay as an inter-operability protocol. The following sections describe the two implementations, one by Sanders and one by Mississippi State University.

4.1.3.1 Sanders Implementation

The Sanders PacketWay implementation focused on developing a high speed, heterogeneous implementation that minimized latency and provided a basis for future implementation of the Secure PacketWay extensions. Figure 5 shows the Sanders heterogeneous testbed configuration. The testbed included HPSC APU nodes, Sun workstations, and Intel PCs. The PCs included two Myricom LANai cards and acted as PacketWay routers – each LANai card served as a half router. Sanders implementation included PacketWay libraries for writing PacketWay applications on the Sun, the PC, and the LANai processor. A generic router for Intel PCs was also implemented. Table 1 shows the performance of the PacketWay router.

Transfer Activity	Sub-Process	Message Sizes			
		10 Word	100 Word	250 Word	500 Word
Network Receive	0 - Allocate SRAM	3.0	3.0	3.0	3.0
	1 - DMA in EEP	1.0	1.0	1.0	1.0
	2 - Determine size	1.5	1.5	1.5	1.5
	3 - DMA in data	0.5	3.0	7.5	15.5
	4 - DMA in trailer	1.0	1.0	1.0	1.0
HR to HR	5 - Allocate SRAM	1.5	1.5	1.5	1.5
	6 - DMA to crypto/HR	2.0	4.5	9.0	16.5
	7 - Cleanup memory	2.5	2.5	2.5	2.5
Network Send	8 - Determine route	1.0	1.0	1.0	1.0
	9 - DMA out route	1.0	1.0	1.0	1.0
	10 - DMA out packet	0.5	3.0	7.5	15.5
	11 - Cleanup memory	2.5	2.5	3.0	3.0
Total Latency		18.0	25.5	39.5	63.0

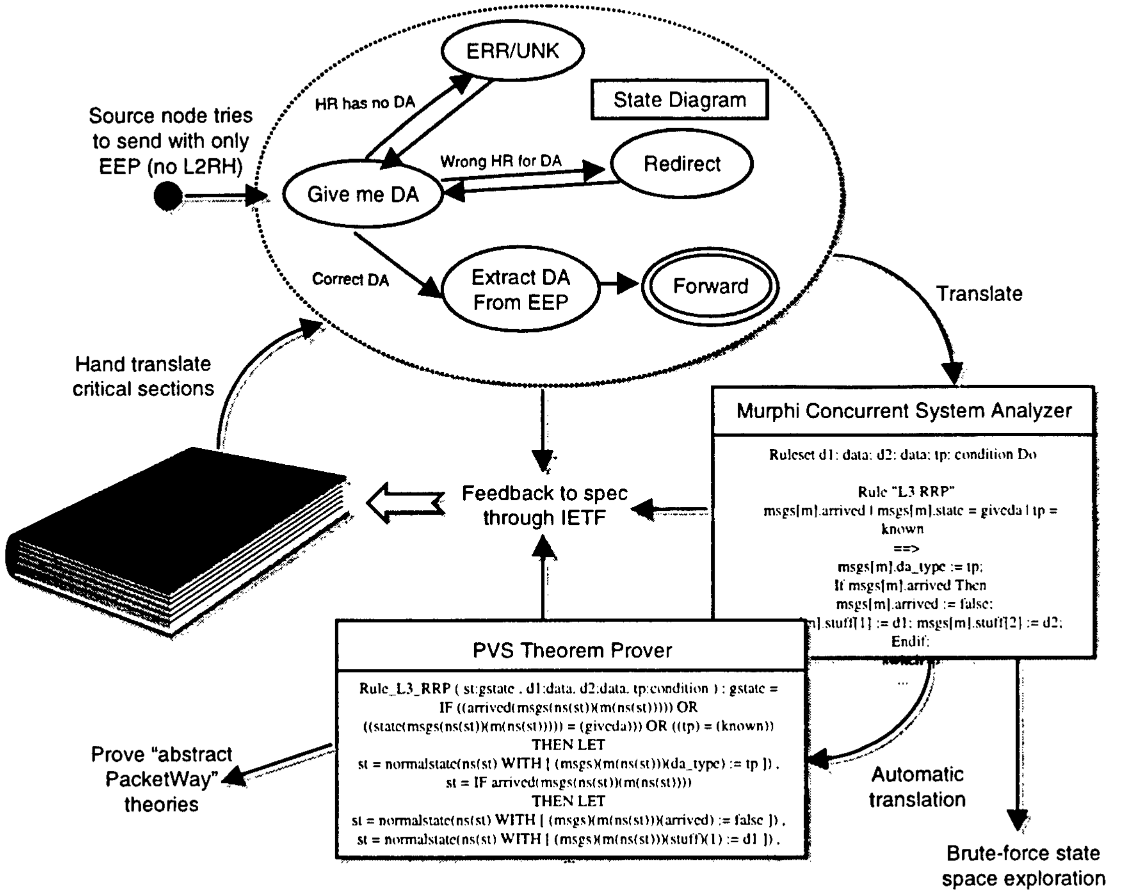
Table 1 Performance of Sanders PacketWay Router (time in us)

4.1.3.2 Mississippi State Implementation

Mississippi State developed a Secure PacketWay implementation that worked over multiple SANs through secure PacketWay routers. Their implementation includes support for multiple transport layers (e.g., Myrinet, UDP, etc.). Another important aspect of the PacketWay protocol provided by Mississippi State was the specification of a PacketWay API that allows applications to be written using the PacketWay protocol.

4.1.4 Formal Methods Verification of the PacketWay Standard

A major advantage of network protocol standards is that their specification promotes multi-vendor implementation. Unfortunately, the text form of these specifications is subject to multiple interpretations, negating the original standardization goals. The goal of applying formal methods to the PacketWay specification was to provide a formal specification and verification of the protocol. Figure 6 shows the process that was used to assist in completing and verifying the correctness of the PacketWay specification. The first step was converting the potentially ambiguous portions of the specifications into state diagrams of two forms: UML (Unified Method Language) and non-deterministic finite state automata. This conversion aided in a first level consistency check, both in finite machine regularity, decidability, reachability, and enhanced peer review, and permits tools to perform verification and potential conversion to a theory based representation. . We used these state diagrams to carry forth simulations (to Murphi) for a brute-force state space exploration and potential deadlock check as well as automatically translate to theories (to PVS from Murphi) for a deeper level consistency check. When specification ambiguities were discovered (e.g. potential lost messages, language clarification, RRP message inconsistencies), these issues were raised to IETF


ERR/UNK
State Diagram
Source node tries to send with only EEP (no L2RH)
HR has no DA
Wrong HR for DA
Redirect
Give me DA
Correct DA
Extract DA From EEP
Forward
Translate
Hand translate critical sections
Murphi Concurrent System Analyzer
Feedback to spec through IETF
PVS Theorem Prover
Prove "abstract PacketWay" theories
Automatic translation
Brute-force state space exploration

```
Ruleset d1: data: d2: data: tp: condition Do

Rule "L3 RRP"
msgs[m].arrived | msgs[m].state = giveda | tp =
known
==>
msgs[m].da_type := tp;
If msgs[m].arrived Then
msgs[m].arrived := false;
m].stuff[1] := d1; msgs[m].stuff[2] := d2;
Endif;
...
```

```
Rule_L3_RRP ( st:gstate . d1:data. d2:data. tp:condition ) : gstate =
IF ((arrived(msgs(ns(st))(m(ns(st))))) OR
((state(msgs(ns(st))(m(ns(st))))) = (giveda))) OR ((tp) = (known))
THEN LET
st = normalstate(ns(st) WITH [ (msgs)(m(ns(st)))(da_type) := tp ]) .
st = IF arrived(msgs(ns(st))(m(ns(st))))
THEN LET
st = normalstate(ns(st) WITH [ (msgs)(m(ns(st)))(arrived) := false ]) .
st = normalstate(ns(st) WITH [ (msgs)(m(ns(st)))(stuff)(1) := d1 ]) .
```

Figure 6 PacketWay Verification Process

through meetings and the mailing list as well as recommended changes to the PacketWay specification.

4.1.5 PacketWay versus IPv6

Some of the questions that frequently arose during the SHARE program were: how do PacketWay and IP relate, which protocol is better, why should PacketWay be used, and are PacketWay (or IP) appropriate for high performance embedded systems? We studied both PacketWay and IPv6 to determine the answer to these and other questions. The paper documenting this comparison is included as an attachment.

4.1.6 PacketWay Video

Mississippi State has developed an overview video of the PacketWay and Secure PacketWay protocol to provide an introduction to the fundamentals of the PacketWay protocol, as well as providing a graphical depiction of source routing and cut-through routing.

The video provides animations of several key PacketWay message types, including L2 source-routing queries and message transmissions.

4.2 MPI

MPI (Message-Passing Interface) is an API for writing portable, parallel programs. MPI was developed by the MPI Forum; a collection of researchers from government, industry, and academia. The SHARE team has participated in various MPI standardization efforts (MPI, MPI-2, MPI/RT) in order to promote the use of MPI in secure, high-performance, embedded, and real-time environments. Under joint funding from the HPSC, ACP, and SHARE programs, an implementation of MPI/RC (resource constrained MPI) for the HPSC APU was undertaken. Figure 7 shows some of the tools available to MPI for HPSC APU developers.

At the outset of the project, a subset definition for resource constrained systems was not available. A subset of functions from MPI-1 was chosen for the initial implementation. As a result of the implementation effort, the SHARE team proposed a subset of functions applicable to resource constrained systems. This proposal resulted in the definition of MPI/RC (MPI for resource constrained systems) being included in the MPI/RT (real-time MPI) document. Figure 8 shows the MPI/RC subset and the portion of that subset currently implemented in the SHARE MPI implementation.

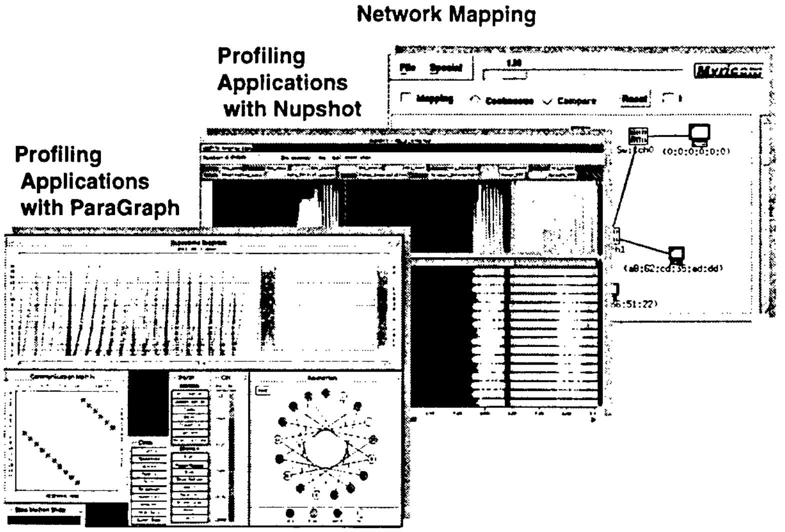
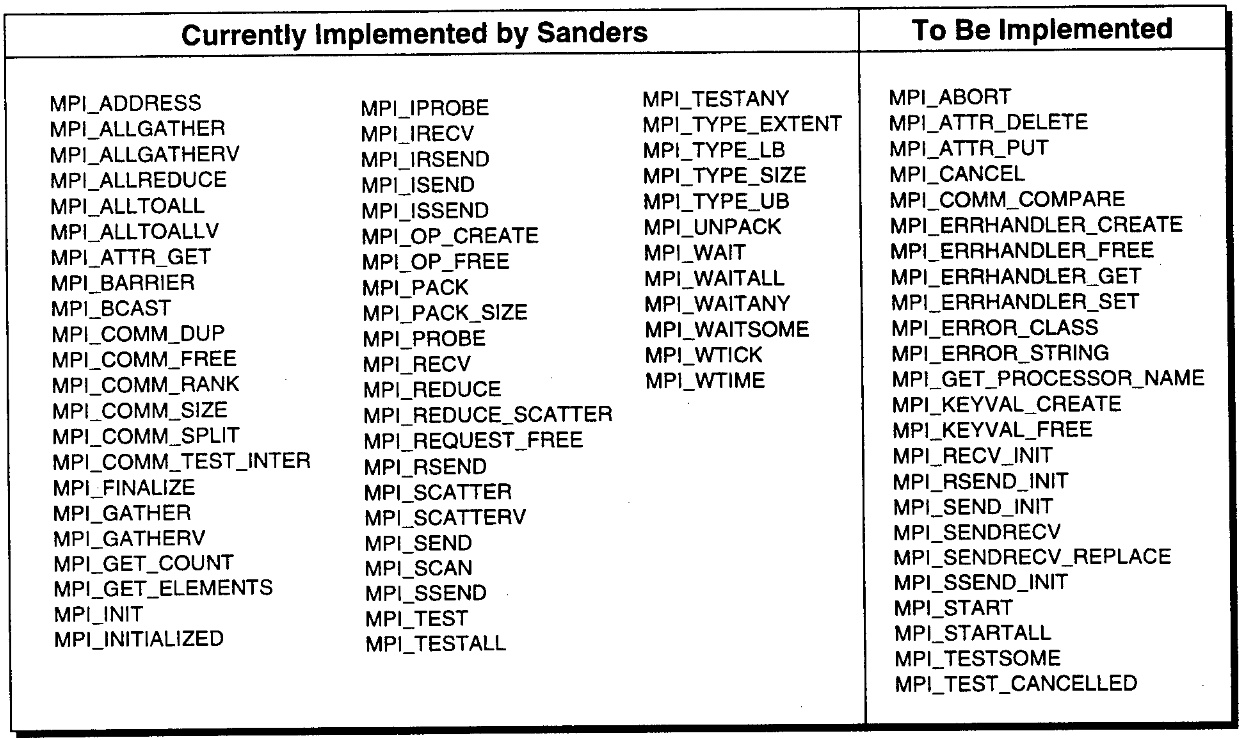

Network Mapping
Profiling Applications with Nupshot
Profiling Applications with ParaGraph

Figure 7 Tools Environment for the SHARE implementation of MPI

A number of applications have been ported to MPI for the HPSC APU including the following:

- Constant False Alarm Rate (CFAR)
- Synthetic Aperture Radar (SAR)
- Sequential Video Phase Processing (SVPP)
- Planar Sub-Array Processing (PSAP)
- Wavelet TCQ Compression
- Super Resolution EO
- High Definition Vector Imaging (HDVI)



Currently Implemented by Sanders			To Be Implemented
MPI_ADDRESS	MPI_IPROBE	MPI_TESTANY	MPI_ABORT
MPI_ALLGATHER	MPI_IRECV	MPI_TYPE_EXTENT	MPI_ATTR_DELETE
MPI_ALLGATHERV	MPI_IRSEND	MPI_TYPE_LB	MPI_ATTR_PUT
MPI_ALLREDUCE	MPI_ISEND	MPI_TYPE_SIZE	MPI_CANCEL
MPI_ALLTOALL	MPI_ISSEND	MPI_TYPE_UB	MPI_COMM_COMPARE
MPI_ALLTOALLV	MPI_OP_CREATE	MPI_UNPACK	MPI_ERRHANDLER_CREATE
MPI_ATTR_GET	MPI_OP_FREE	MPI_WAIT	MPI_ERRHANDLER_FREE
MPI_BARRIER	MPI_PACK	MPI_WAITALL	MPI_ERRHANDLER_GET
MPI_BCAST	MPI_PACK_SIZE	MPI_WAITANY	MPI_ERRHANDLER_SET
MPI_COMM_DUP	MPI_PROBE	MPI_WAITSOME	MPI_ERROR_CLASS
MPI_COMM_FREE	MPI_RECV	MPI_WTICK	MPI_ERROR_STRING
MPI_COMM_RANK	MPI_REDUCE	MPI_WTIME	MPI_GET_PROCESSOR_NAME
MPI_COMM_SIZE	MPI_REDUCE_SCATTER		MPI_KEYVAL_CREATE
MPI_COMM_SPLIT	MPI_REQUEST_FREE		MPI_KEYVAL_FREE
MPI_COMM_TEST_INTER	MPI_RSEND		MPI_RECV_INIT
MPI_FINALIZE	MPI_SCATTER		MPI_RSEND_INIT
MPI_GATHER	MPI_SCATTERV		MPI_SEND_INIT
MPI_GATHERV	MPI_SEND		MPI_SENDRECV
MPI_GET_COUNT	MPI_SCAN		MPI_SENDRECV_REPLACE
MPI_GET_ELEMENTS	MPI_SSEND		MPI_SSEND_INIT
MPI_INIT	MPI_TEST		MPI_START
MPI_INITIALIZED	MPI_TESTALL		MPI_STARTALL
			MPI_TESTSOME
			MPI_TEST_CANCELLED

Figure 8 MPI 1.2 Resource Constrained

Although these algorithm ports have not been performed under SHARE funding, they do demonstrate the usefulness of the MPI implementation for developing applications for heterogeneous systems. The following are some of the lessons learned from the experience of porting these applications:

- Porting went smoothly and has in general taken less time than expected.
- Applications that overlap communication and computation perform well.
- Optimizations for the SHARC are needed to obtain the best performance.
- Limited program memory on SHARCs has been a problem at times.

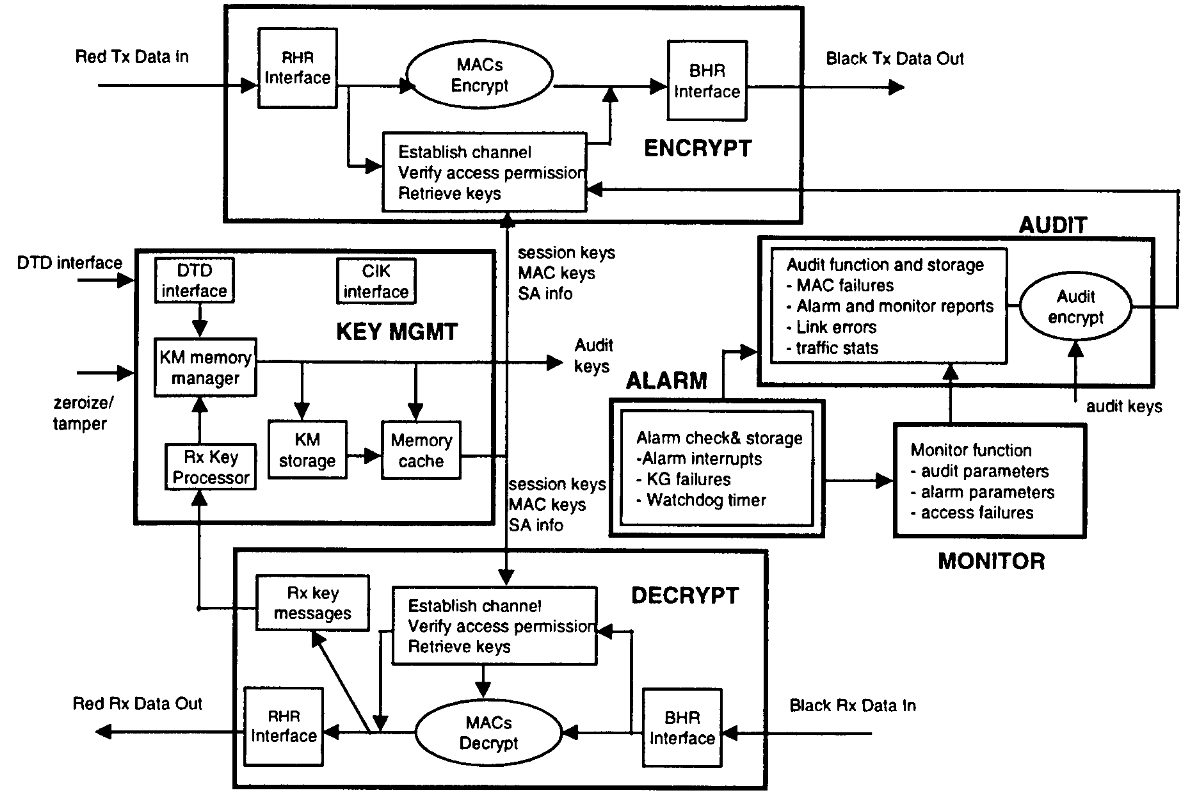


Figure 9 SHARE Infosec Module Design

4.3 Secure Architecture Design

One of the key accomplishments of the SHARE program was the definition of a secure network architecture based on Secure PacketWay.

A detailed design for a DES-based SHARE cryptographic module (SIM) was developed. To date, four SIM boards have been assembled.

An implementation of the SHARE security architecture was developed. This development began by implementing the security extensions to PacketWay. The implementation progressed to a prototype implementation of the SHARE network security architecture. This prototype interfaces to the software encryption module API and encrypts and decrypts all traffic through the secure PacketWay routers.

The SHARE team presented interim SHARE results, including Network Security Architecture and results of the Secure PacketWay development to a panel at the National Security Agency. The NSA review was favorable and included feedback concerning additional authentication which was added to the implementation. As a result of this meeting, NSA has fed back a PARADE data sheets and FASTLANE specifications that will help us in our Type I cryptography design specification considerations.

Additional accomplishments include the completion of a comparison study of the security features provided in Secure Packetway, IPv6, and Space Communications Protocol

Standards (SCPS) and the definition a generic, network protocol independent, message passing interface between the SIM and the SHARE router.

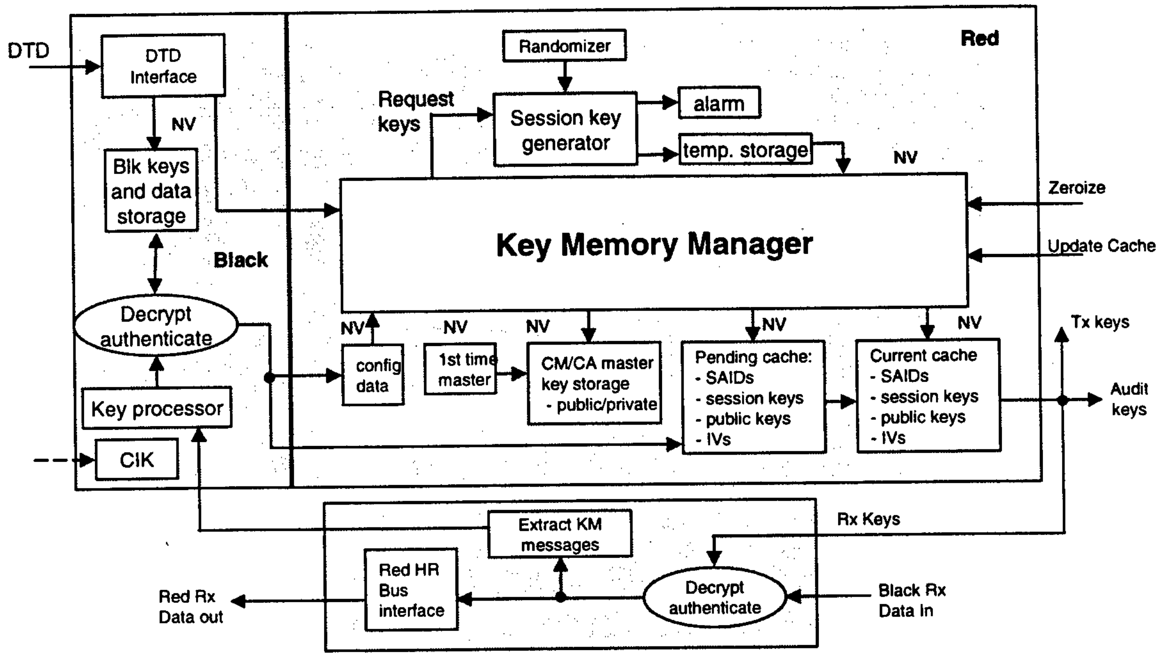


Figure 10 SHARE Key Management

A Key Management Plan was constructed in conjunction with the detailed design. A block diagram as seen in Figure 10, describes the design.

4.4 Network Simulation

A basic infrastructure for ptolemy-based Secure Network Simulation was developed with the generation of primary network element "stars." These stars include "DEHost," "DELANai," and "DESwitch." Additionally, Router and HalfRouter "stars" and "galaxies" have been generated. Currently, small networks, called "universes" in the Ptolemy realm, can be built from these network elements, i.e., "stars" and "galaxies." Figure 11 shows the currently available simulation elements.

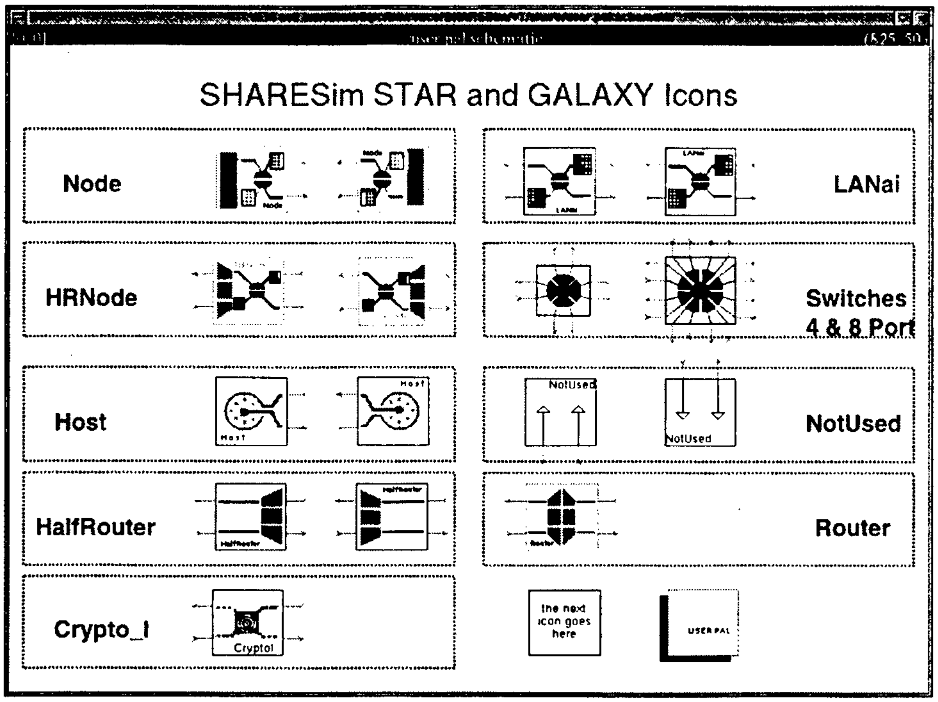


Figure 11 Ptolemy Simulation Objects for the SHARE Simulation

4.5 Framework for Developing Communication Software for Two-Level Multi-computers

One of the main problems we found when developing software for a two-level multi-computer is the lack of a common framework for developing code. For example, this problem is manifested throughout the community of researchers developing code for Myricom network interfaces. In order to address this problem, the SHARE team began looking at providing a framework for writing protocol agile communication software for two-level multi-computers. A prototype implementation of a communication framework was developed during which time we discovered a standards effort for intelligent I/O (a.k.a. I_2O). The I_2O effort involves such companies as Intel, Microsoft, 3Com, etc. and will likely become a very important and ubiquitous standard. The I_2O standard provides a software framework for developing intelligent I/O systems. It allows drivers for I/O systems to be portably written and extended. An example use would be augmenting a network I/O device with code that serves to accelerate MPI. The MPI driver could be used in any number of I/O devices since the I_2O specification provides a common API for such devices.

A source level LANai debugger was written during the course of this program, thereby speeding the development of LANai-based system software for the SHARE program. This debugger allows the developer to probe the control registers and general status of the LANai 4.0 processor from the host platform as well as single stepping through user code, setting breakpoints, etc. The LANai debugger proved to be an important tool during both the development and testing phase of the SHARE program.

4.6 *Standards Participation*

The SHARE team has been actively working with the IETF PacketWay working group to develop the PacketWay specification. In particular, the team participated in discussions regarding changes to PacketWay draft specification that supports security. A draft of the PacketWay Specification has been revised by IETF working group based on SHARE team inputs and has been approved based on an IETF review.

Security extensions to PacketWay were developed. The goal of these extensions was to non-intrusively extend the base PacketWay protocol to support the security architecture of SHARE, without hindering the implementation of other potential security architectures. These extensions are described in a document submitted to the IETF for inclusion as an official part of the IETF.

The SHARE team supported the MPI Forum working group meetings with the development of MPI-2 extensions for security and real-time. The final MPI-2 standard did not include security and real-time extensions. The SHARE team has participated in the MPI/RT Forum (a forum explicitly formed to address real-time and embedded systems) which was formed shortly after the MPI-2 process ended.

4.7 *Documentation*

An application survey summarizing the secure requirements of various programs was reported during SHARE quarterly reviews. The following is a list of documents created as a result of the SHARE project.

System Requirements
Network Security Architecture
Secure PacketWay Specification
Comparison of PacketWay and IPv6
Crypto Protocol Study
Network Simulation Plan
Key Management Plan

These documents are attached to this report as an appendix.

5 Conclusions/Recommendations

The SHARE program has led to the following conclusions/recommendations:

1. The PacketWay standard provides a protocol for inter-operably connecting high performance SANs. Currently, there are few organizations that have embraced the PacketWay standard. For this standard to be a success, participation from others should be and must be encouraged.
2. The MPI standard has been shown to be an appropriate way to develop high performance, portable applications on embedded systems. The results of the MPI/RT Forum's attempt to providing real-time and embedded extensions to MPI are particularly important to how future high performance embedded systems will be developed.
3. Although MPI allows portable, parallel applications to be developed. MPI only specifies the API, not the underlying protocols. This makes inter-operability between

different vendor implementations difficult. An inter-operability forum (Interoperable MPI – IMPI) has been created to address this problem.

4. Advancement of standards such as PacketWay and MPI is of utmost important in providing secure, high-performance, inter-operable embedded systems. Members of the SHARE team are committed to continued participation in such standards.
5. L-3 has built several prototype cryptography modules as a result of the SHARE program. The development of software for these modules and the integration of these modules into a testbed has not been completed.
6. One of the key improvements that can be made in the development of software for two-level multi-computers is the advancement of a common framework for developing software. Such a framework would be generally useful to developers that work with intelligent I/O systems. The I_2O standard provides such a framework and may potentially revolutionize the way software is written for I/O devices.

November 1997 Business Status Report

Final Report

to

Rome Laboratory
Agent for ARPA

for

Cooperative Agreement F30602-95-2-0051
SHARE-HPSC

Prepared by
Sanders, A Lockheed Martin Company
Signal Processing Center, PTP02-B002
Advanced Engineering & Technology Division
PO Box 868
Nashua, NH –03061-0868

30 January 1998

1 Cost Performance

The charts below depict the program cost status as of the end of the contract (November, 1997). Chart (a) details the cumulative costs, planned and actual incurred, under the DARPA contract funding. Chart (b) shows the cumulative costs, planned and actual, incurred under Sanders cost sharing activities. Chart (c) combines the data in the previous two charts to show total program cumulative cost performance.

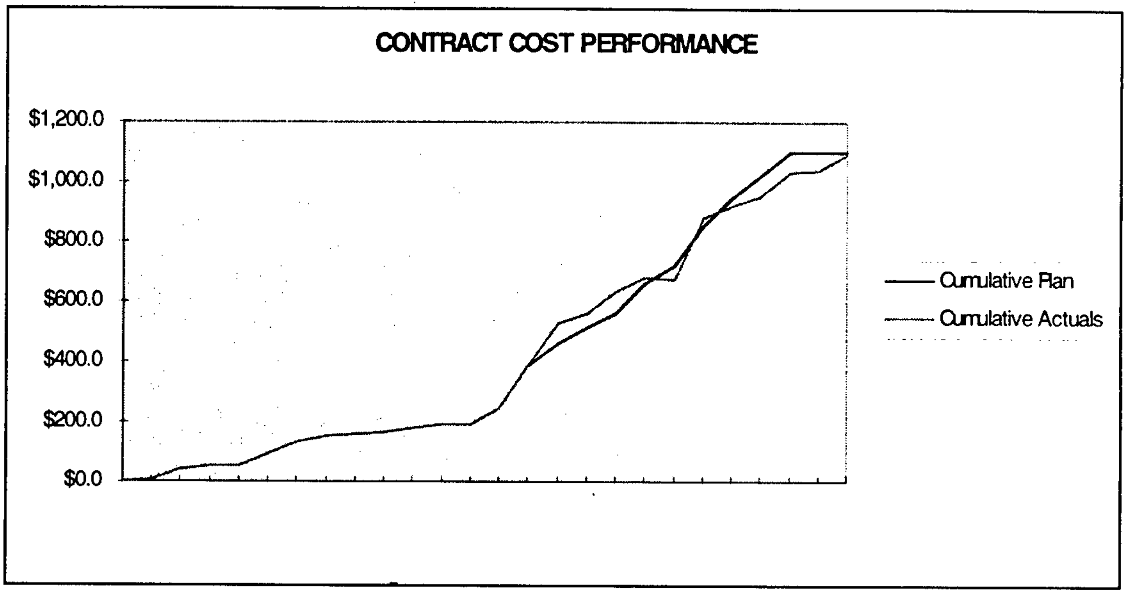


Chart A **Contract Cost Performance**

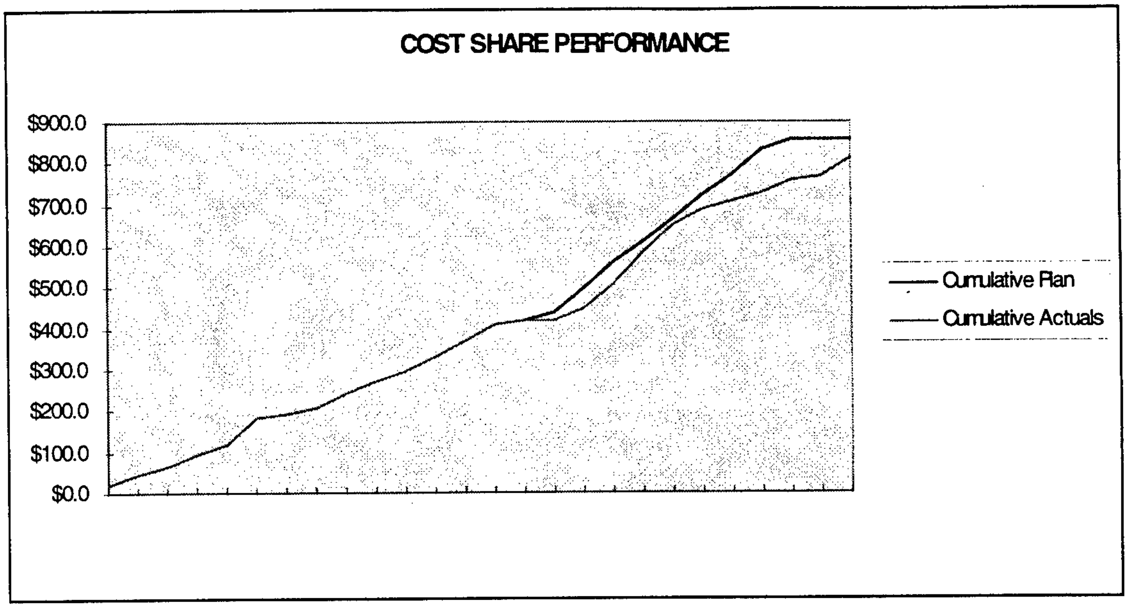


Chart B Cost Share Performance

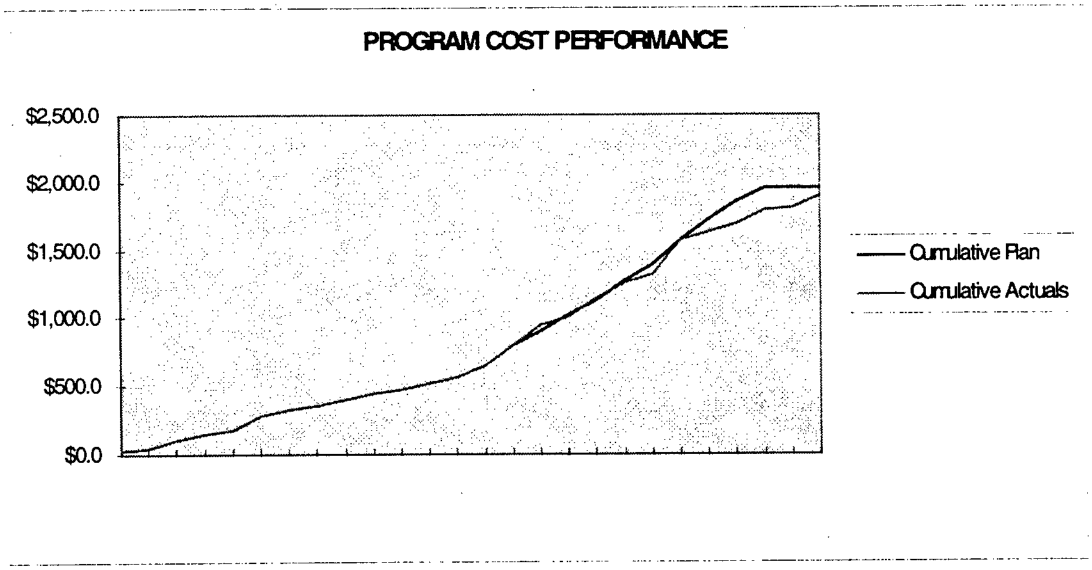


Chart C Program Cost Performance

SHARE•HPSC
System Requirements

Rev –
5 Sep 1996

Approved By:	For:	Date:
Jeff Smith	Sanders	9/5/96
Greg Byrd	MCNC	9/5/96
Robert George	MSU	9/5/96
Jack Brizek	Lockheed Martin Communication Systems	9/5/96

TABLE OF CONTENTS

1. ABSTRACT

This document describes the technical requirements for SHARE•HPSC*, a fully integrated network environment that links System Area Networks (SANs) of any design, and provides high speed, low latency, multi–level secure information transfer between SAN hosts. SHARE allows authenticated SANs of differing types and security levels to communicate at 1.28 Gb/s rates (2.56 Gb/s full duplex) over a public network.

SHARE is an open system utilizing Secure PacketWay as the network packet switching standard to connect and accomplish information transfer between smaller high performance local SANs. SANs may transmit and receive data from peer SANs of like security level, and SANs of higher security level may read data from lower level SANs. These data transfers are permitted when they are consistent with SAN authentication levels, specifically granted security authorizations, and on–line verification of the security integrity of the network. A SAN host node can range from a single workstation or personal computer to an embedded special purpose scalable multiprocessor.

Development of the SHARE environment begins with Phase 1. As part of Phase 1, the system architecture and key hardware designs are developed. Simulation models are created and exercised to verify predicted network performance and to evaluate equipment tradeoff alternatives. Three demonstrations are defined to show cumulative technical progress. The demonstrations are hardware and software based simulations of the developing SHARE environment. The use of unclassified available COTS hardware is encouraged.

A major goal of the SHARE development program is to enable interested companies to insert this technology into products needed by the commercial sector, and by the government sector where NSA plays an integral support and certification role.

* Secure Heterogeneous Application Runtime Environment for High Performance Scalable Computing

2. DOCUMENT SCOPE

This document contains three major technical sections, i.e. SHARE (1) System Overview, (2) Phase 1 Requirements, and (3) Final Implementation Plan. The Overview section 3 presents SHARE system concepts. The Phase 1 section 4 describes the technical requirements for the Phase 1 development. The Final Implementation Plan section 5 describes additional aspects of the SHARE environment which are planned in a succeeding phase or in the final design.

There are two kinds of Phase 1 requirements defined herein, i.e. those required of the SHARE design, and those required of demonstration models and hardware. In general, analyses, tradeoffs, performance projections, simulations, and architectural and equipment design are to be against the Final Implemented design. The demonstration models provide simulations of performance of the final implementation, however, the building of exact verification hardware is not a part of this effort. These models are comprised of generally available hardware and software (COTS), adequately modified to evaluate the performance parameters of interest.

The following documents further detail the concepts presented herein.

1. "SHARE•HPSC Network Security Architecture" (released).
2. "SHARE•HPSC 'Dogwood' Cryptographic Module" (in preparation).
3. "SHARE•HPSC Cryptographic Key Generation–Distribution–Management" (planned).

3. SHARE•HPSC SYSTEM OVERVIEW

3.1. DEFINITION OF TERMS

A definition of terms is provided in Section 6. to aid in the readability of this document. The group of terms as a whole also provides insight into concepts which drive the SHARE•HPSC architectural development. Five terms are included here which explain the SHARE•HPSC acronym.

Security: In the SHARE environment, cryptographic methods and open standards are used to ensure information survivability in the transmission of (multi–level) secure packets of information over a public network. The network security objectives are data integrity, confidentiality and authentication, and availability.

Heterogeneity: SHARE is the hardware and software fabric that provides interoperation between inherently dissimilar network components, i.e. between SANs of various types and security levels, and between host nodes having differing bandwidth and operating system requirements.

High Performance: High performance means high bandwidth, low latency, operation in an error environment, on–line and off–line testability, and controlled fault recovery.

Application Runtime Environment: The provided high speed, low latency network environment enables applications to run across cooperating heterogeneous SANs in near real time. Application programs use traditional Application Program Interface (API) software to access the underlying network services.

Scalable: Scalability is the ability to increase or decrease the size and heirarchy of a connected network of SANs using fundamental SHARE•HPSC building blocks. Very large scale extensions to the SHARE network of SANs are possible using basic building blocks which include, but are not limited to, secure and non secure routers, local SANs, and high speed public network switches.

3.2. SHARE CONCEPTS

The concepts presented in this section are included in the SHARE Final Implementation as described in Section 5. The elements to be implemented in the Phase 1 effort are described in Section 4.

3.2.1. An Integrated Standard

As described herein, the SHARE specification is baselined on the use of the Secure MPI standard (for the host interface) and the Secure PacketWay standard (for the SAN/network interface). These open standards provide well defined interfaces and promote interoperability between multiple vendor products well into the future. The integration of security services within the SHARE environment ensures the elimination of compatibility problems associated with non integrated products, and the elimination of the need to re–evaluate and re–certify the security of product extensions to non integrated implementations.

3.2.2. Public vs Secure Data Transport

Network architectural requirements herein can be partitioned into two broad categories, (1) those defining the *public* aspects of data transport from source to destination, and (2) those defining the *secure* aspects of the data transport. These two categories are not independent. In order to transmit secure data over a public network, the data format which enables the security functions to be implemented (headers, trailers, tags, keys, data, authentication, etc.) must 'fit' into, or be encapsulated by, the format of the public carrier. The SHARE environment is based on the utilization of standardized public interfaces, i.e. MPI for SAN host nodes and PacketWay packet network protocols for the SANs. The necessary security encapsulations (extensions), Secure MPI and Secure PacketWay, shall be integrated into the corresponding standards.

3.2.3. Secure Data Concepts and NSA Certification

In the following paragraphs, concepts associated with SHARE secure data transport are presented. The concepts for public data transport are inherently standard networking industry practice and are widely accepted. Therefore, the public data transport topic is not expanded in great detail. In those cases where the security discussion requires specific public transport details, they are provided.

In general, this document presents requirements in the context of a commercial application. Requirements are also included herein which address government application, and which when implemented are expected to result in NSA certification. However, requirements of a design certified by NSA are specific and elaborate. When met, they serve to authorize the design for government application at the specific certified security level. These specific requirements are contained in classified documents which cannot be repeated here. It is intended that in the final SHARE implementation, the SHARE Network Trusted Computing Base (NTCB), the Network Manager and Network Operating System (NOS), and the hardware and software components which make up the SHARE network will be proven compliant to the government's B2 "Mandatory–Structured Protection" (TCSEC – Trusted Computer Security Evaluation Criteria) level.

3.2.3.1. Security: Total Effects

System security is provided by the total effect of all system elements associated with the information transmission. Three characteristics can be associated with computer security, i.e. data integrity (including accuracy and authenticity), data confidentiality (including access controls and encryption), and availability (including reliability and maintainability). These characteristics include the effects associated with an operator logging onto a system workstation with a given password, to the effects associated with encrypting data with an algorithm and keys designed to provide a particular level of security. The following paragraphs identify from a top level the protection provided by SHARE against threats to information survivability. The reader should read this entire document to understand the extent of the protection provided.

3.2.3.2. Protection Against Threats to Data Integrity

Assuming that the data transmission system design is compatible with the transmission medium's error rate, the five most common threats to data integrity are:

1. Humans
2. Hardware malfunctions
3. Network malfunctions
4. Logical problems
5. Disasters

Human threats, item 1 above, may occur inside or outside the secure perimeter defined by the 'secure room' in which the humans work. Inside threats are countered by the security policy of the local

SAN. Also protected by the 'secure room' is the SHARE secure router which interfaces each SAN, which contains the cryptographic function, and which converts red to black data (and vice versa). Threats outside of the secure SAN perimeter, i.e. against black data outputted from, and inputted to, the SAN, are countered by cryptographic functions centered in the cryptographic module of the secure router.

SHARE protects against data integrity compromise associated with the problems of items 2 through 5 via extensive on–line and off–line test functions. The on–line functions immediately identify the occurrence of real time problems via alarms which warn system users of the anomaly (or potential anomaly) and which immediately prevent any further classified data transmission. On–line functions also continuously establish the network's ability to transfer data when requested. The off–line testability functions facilitate the equipment's timely repair and equipment and network restart. SHARE on–line and off–line testability features are addressed in System Testability, section 5.4.

3.2.3.3. Protection Against Threats to Data Confidentiality

A given system which is designed to protect the integrity and privacy of its data may be subjected to the following basic types of security threats:

1. Physical (theft, spying, false ID, dumpster diving)
2. Wire based (eavesdropping, modem dial in, machine impersonation)
3. Service authentication (password traps, password cracking, password algorithm weakness)
4. Programming (virus, code bombs, trojan horse, firmware and OS updates and downloads)
5. System loopholes (port piggybacking, bypassing services, reconfiguration and initialization glitches)

Since the local SAN and the SHARE router are contained within the secure room boundaries of the local SAN, physical and wire based threats, items 1 and 2 above, must attack outside of this boundary against SHARE black (protected) data. These threats are against the extensive cryptographic measures provided by SHARE. Service authentication (Item 3) originates with basic password protection provided the local SAN. Programming and system loopholes (items 4 and 5) are basically software attacks. To ensure that such threats will not be successful, the SHARE Final Implementation software shall be subjected to rigorous planning, design, review and examination, testing, and pre–installation 'break in attempts'.

3.2.3.4. Protection Against Threats to Availability

A secure computer environment must keep its resources available to its users. In SHARE, the authenticated SAN enclaves are secure and trusted at the security level defined for that SAN. Outside of the secure SAN enclaves, SHARE data is black and information transfer is via public network. Threats to system availability are then by definition against the public network. SHARE enables protection against denial of service by a particular network by providing the means for connection to and through any network. For example, a local SAN may connect to the network via different (more than one) routers which interface to different SANs. Also, if the primary route through the mesh (as provided by the primary routing table) is unavailable, alternative routing tables are used to circumvent links that are unavailable. In addition, the SHARE Final Implementation provides on–line and off–line testability features which address availability and maintainability, i.e. see System Testability, Section 5.4.

3.2.3.5. SAN Authentication

To accomplish secure communication, a SAN routes information through a SHARE authenticated router (or equivalent) which recognizes security protocols enabling multilevel secure channels. Authentication means that a router/SAN has been installed into the SHARE environment and that the SAN's security context and other characteristics are available to other SANs of the same and higher security levels. A SAN may transmit/receive from other authenticated SANs of the same security level, and also read from authenticated SANs of lower security level. These authenticated routers thus represent trusted firewalls for the host SANs which they serve.

3.2.3.6. Access Control

SHARE controls access to information such that only properly authorized individuals or processes have access to the information. Six fundamental requirements of the SHARE system derive from this basic objective – four deal with what is needed to control access, and two deal with how to obtain credible assurances that this is accomplished. These six requirements are described below and are further amplified in the "Orange Book", "Trusted Computer System Evaluation Criteria (TCSEC)".

> Note: The "Red" book, "Trusted Network Interpretation", provides interpretations of the "Orange" book for trusted computer/communications network systems. The "Red" book thus extends the TCSEC "Orange" book to networks of computers and describes a number of additional security services that arise in conjunction with networks.

Policy:

1. Security Policy: There shall be an explicit and well defined security policy enforced by the system, i.e. a set of rules used by the system to determine whether a subject is permitted to gain access to a specific object.
2. Marking: Access control labels that identify an object's security level shall mark each object.
3. Identification: Individual subjects must be identified and authorized as to the class of information that they may deal with.
4. Accountability: Audit information must be selectively kept and protected so that actions affecting security can be traced to the responsible party. Audit information must also be protected.

Assurance:

5. Assurance: Hardware/software mechanisms shall be provided that can independently evaluate and provide sufficient assurance that the system enforces requirements 1 through 4 above. These mechanisms are typically embedded in the operating system.
6. Continuous Protection: The trusted mechanisms that enforce these basic requirements must be continuously protected against tampering and/or unauthorized changes.

To accomplish these six objectives, SHARE data shall be provided with network security elements which accomplish privacy, multilevel security access control, authentication, non–repudiation, key management, spoofing protection, and audit functions.

3.2.4. Scalability

SHARE provides network scalability via the repeated application of its major building blocks. A Network Manager manages the building blocks. The Network Manager is a hardware/software network component that has access to the SHARE elements within the network and provides the network services that enable the building blocks to work as an integrated environment. The three current major building blocks are:

1. The secure router, which includes the cryptographic functions
2. The various types of SANs
3. The various types of inter SAN public networks, e.g. Myrinet, etc.

The secure router (item 1) may interface to a particular trusted SAN (item 2) via any number of hardware variants, e.g. PCI cards for a PC implemented LAN interface, SBus adapters for Sun workstations, Myrinet boards and LANai interface chips for HPC platforms, etc. The router also provides the interface to the untrusted public network (item 3) and its required protocols. Each public network is interfaced by the appropriate network interface unit (bridge, router, gateway) which provides the compatible physical, data link, and network layer protocols. Public network switches provide the link ports for connection to the local SANs via the SHARE authenticated routers.

In addition, the trusted firewall and cryptographic functions are integrated into the secure router.

The Network Manager is located in a secure room at a security level that is equal to that of the highest data security level to be protected.

3.2.5. Heterogeneity

Each SAN is considered a homogeneous entity, i.e. all nodes are of like security level and interface via the implemented SAN local network protocol. When two dissimilar SANs communicate, they do so via a common network protocol. SHARE provides this common network protocol (Secure PacketWay) and this protocol is embedded as the top layer in the secure router. The fact that such heterogeneous local networks are involved in the information transfer is transparent to the user.

3.2.6. The Network Manager

The Network Manager and its associated secure operating system is the performs the administration function of the SHARE network. To ensure network integrity, the manager monitors network operation, monitors and manages the elements in the net (routers, gateways, etc.), and reacts to disturbances in these elements (security related and otherwise). Likewise, each of the elements in the net also reacts to disturbances in the operation of the Network Manager. The Network Manager includes network components which are distributed throughout the network, and also which are centralized. The centralized element includes a console and display for a human administrator to observe net performance and to update network parameters as necessary, i.e. access control, security levels, authentications, need to know authorizations, key contexts, etc. The diagram of Figure 1 includes a Network Manager element. Paragraph 5.1.1., item 3.,lists a top level set of requirements.

3.2.7. SHARE Network Structure

SHARE design is based on a network of point to point links that can be configured as any of several topologies, i.e. tree, star, mesh, etc. An example SHARE network structure is illustrated in Figure 1. Local SANs of various security levels are connected via routers to the high speed public network. The public network operates via a PacketWay network layer protocol. An extension to the PacketWay protocol is to be developed which incorporates the SHARE security provisions. The public network of choice is Myrinet which provides the physical and data link layers, and which has a data transmission bandwidth of 160 MB/s (1.28 Gb/s) and a full duplex transmit/receive bandwidth of 320 MB/s (2.56 Gb/s). For the classified SANs, the associated routers are Secure SHARE/PacketWay routers. Standard routers, or SHARE/PacketWay routers without the cryptographic module, connect the unauthenticated SANs. Additional SHARE boxes, e.g. bridges, gateways, are intended as the need develops. The SHARE Network Manager and the Network Operating System facilitates network operations via associated controls and displays.

A functional overview of the SHARE•HPSC network is provided in the "SHARE•HPSC Network Security Architecture" document.

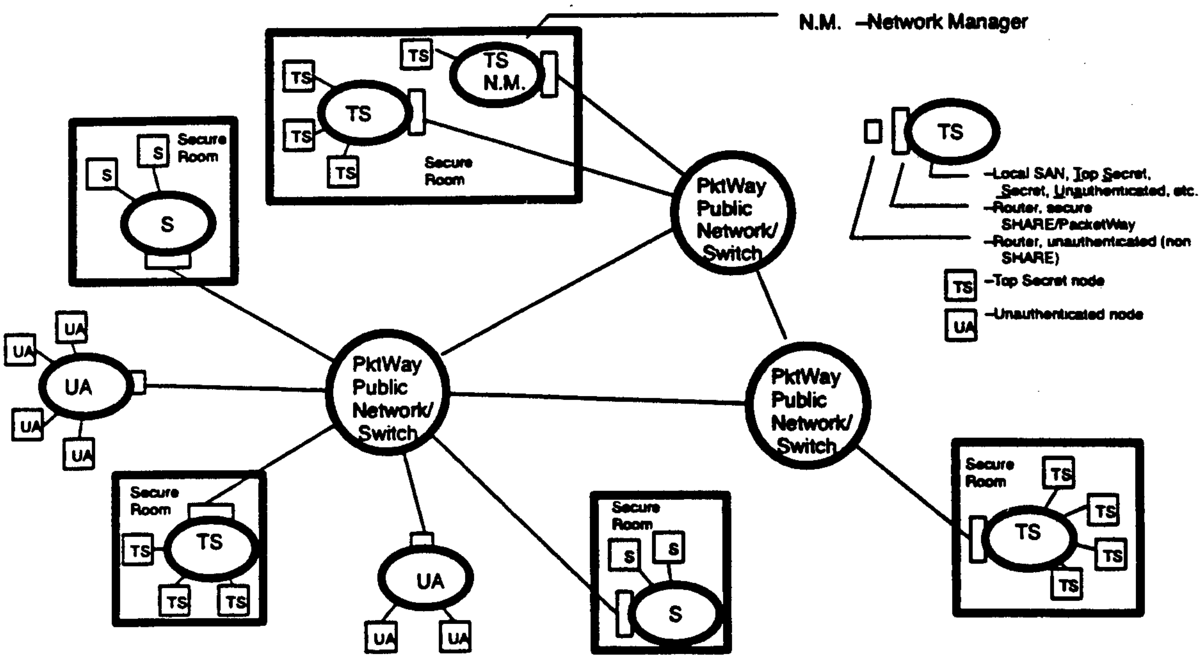

N.M. –Network Manager
TS
N.M.
TS
TS
TS
TS
TS
Secure Room
Secure Room
S
S
S
PktWay Public Network/ Switch
PktWay Public Network/ Switch
PktWay Public Network/ Switch
–Local SAN, Top Secret, Secret, Unauthenticated, etc.
–Router, secure SHARE/PacketWay
–Router, unauthenticated (non SHARE)
TS –Top Secret node
UA –Unauthenticated node
UA
UA
UA
UA
UA
Secure Room
TS
TS
TS
TS
UA
UA
UA
Secure Room
S
S
S
Secure Room
TS
TS
TS
TS
TS

Figure 1: An Example SHARE•HPSC Network Structure

3.2.8. Phase 1 Proof of Concept

For Phase 1 proof of concept, the SHARE secure router shall use the Myrinet source routing high speed data link layer protocol. Myrinet operates at the physical and data link layers. The network layer protocol, PacketWay, is a network packet service which is layered onto Myrinet. SHARE shall employ the source routing (L2) capability of PacketWay. Migration to a strategy which also uses the destination routing (L3) capability of PacketWay shall be provided.

Communication to/from a Myrinet network operates via a LANai interface which incorporates the Myrinet Control Program (MCP). The LANai is a high speed integrated circuit specifically designed to provide this interface. MPI provides compatibility with the security policies used by the router. It is important to note that Secure MPI is not required to have trusted software. The security function is entirely incorporated into the cryptographic function in the secure router.

4. SHARE•HPSC PHASE 1 REQUIREMENTS

This section presents the requirements for the SHARE Phase 1 development. The goals for Phase 1 are (1) to define the elements in the SHARE environment (architectural design), (2) to verify that current technology is able to support its key requirements (trade off studies, simulations, demonstrations), and (3) to further develop elements which are the foundation of the SHARE network fabric (equipment and module design). Note: It may help the reader to first read section 5., SHARE*HPSC Final Implementation Plan, to gain a better understanding of the overall SHARE design.

4.1. SUMMARY: PHASE 1 OUTPUTS

The following is a summary of the output goals of the Phase 1 development. Each output is to be used as the input to a subsequent step in the development of the SHARE Final Implementation. The requirements for these elements are amplified in subsequent paragraphs.

1. –A network 'Final Implementation' architectural design, i.e. a High Performance Computing (HPC) low cost scalable embedded architecture, shall be provided. Related research, analysis, and tradeoff studies shall be referenced.
   –A system hardware architecture model shall be presented as a result of the analyses.
   –All objects that are required, hardware and software, shall be identified and associated with a network location.
   –All necessary interfaces and which hardware and software objects require additional development, shall be indicated.
   –The architectural design shall indicate how security is accomplished and shall indicate the identity, location, and description of the elements of the security design.
   –The security design shall include MLS provisions, key management methods, and on-line and off-line test features appropriate for B2 (TCSEC) certification.
2. A network 'Phase 1' architectural design shall be provided. The relationship between this design and the 'Final Implementation' design of item 1. shall be indicated.
3. Extensions to PacketWay and MPI, i.e. Secure PacketWay and Secure MPI, shall be developed which provide for security features required of a B2 network design. There shall be efforts to incorporate the extensions into the base standards.
4. A system level software architecture design shall be provided, and software shall be delivered which supports the Phase 1 development and its demonstrations.
5. The design for a secure router shall be provided. The design shall be to the hardware schematic level and the software module identification level.
6. The design for a Type 4 cryptographic module to be included in the secure router of item 4. shall be provided. The design shall be to the hardware schematic level and the software module identification level.
7. The design for a Type 1 cryptographic module to be included in the secure router of item 4. shall be provided. The design shall be to the hardware schematic level and the software module identification level.
8. Trade off studies of key architectural and equipment design decisions shall be provided.
9. Simulations of network performance shall be provided. The relationship between the simulations and the 'Final Implementation' design of item 1. shall be indicated.

10. Demonstrations shall be provided to assess progress. The relationship between the demonstrations and the 'Final Implementation' design of item 1. shall be indicated.

4.2. SYSTEM / SECURITY DESIGN

4.2.1. System Architectural Requirements

The Phase 1 design goals shall be consistent with a high performance, connectionless oriented, secure packet switching network. The basic system architectural needs are:

1. Performance: The primary operating parameters that shall be maximized are high bandwidth, low latency, and reliable operation in an error environment. The Myricom HPC secure network connectivity of heterogeneous nodes (to include embedded HPSC, unix workstations and Power PCs) shall achieve a sustained network bandwidth of 160 MB/s, a full duplex bandwidth of 320 MB/s.
2. Network throughput is of primary concern. The design and estimates of performance shall take into account:
   - Algorithm(s) execution and complexity
   - Communication link parameters, the number of allowable hops, alternative routing strategies, allowable packet lifetimes
   - Trusted and untrusted path delays, including firewalls, including 'busy path' avoidance alternatives
   - Hardware and software latency
   - Hardware and software modularity 'penalties'
   - Scalability 'penalties' incurred by the extendable network
   - Multiple security levels
3. SAN independence: There shall be nothing in the SHARE•HPSC design that precludes its connection to a SAN of any type from any vendor.
4. Guidelines: The design of the SHARE system and Secure PacketWay shall use the TCP/IP protocols as guidelines for the services that are to be provided. However, it is to be understood that to achieve the high speed, low latency performance required by SHARE cooperating SANs, some streamlining and reduction of TCP/IP services which unduly restrict performance is expected.

4.2.2. Specific Security Requirements

The following specific TCSEC security aspects shall be implemented in the Phase 1 design:

1. Multilevel security protection shall be supported. For Phase 1, single level protection shall be implemented. Each local SAN shall be interfaced to the SHARE network by a secure router (or equivalent) that manages the security of the data from/to the local SAN. Transmitted data from a SAN must be at the security classification of that SAN. Received data allowed to pass to a SAN shall be of that SAN's security level or lower. Note: In the planned Final Implementation, the secure routers shall be monitored and managed by the Network Manager and the NOS.
2. The secure transmission and reception of data by a host is to be accomplished within the framework of the following protocols:
   - Secure MPI (developed by MPI Forum MPI–2 working group),
   - Secure PacketWay (incorporated by the IETF PacketWay working group)

3. Identify and justify by research and analysis the optimal placement of security hardware. Present a system security architecture model as a result of this analysis.

   The SHARE network design shall provide for cryptographic unit placement considering the capabilities of high performance computing. The contractor shall utilize NSA security approval guidelines to evaluate the resulting network security architecture. The architecture and security policy implemented by SHARE shall reflect:

   1. Participation in the IETF PacketWay working group, resulting in a proposed draft standard for security enhancement to the PacketWay protocol.
   2. Participation in the MPI Forum MPI-2 working group, resulting in a proposed. draft standard for real time and security enhancements to MPI.
   3. Documentation available via the National Computer Security Center (NCSC), part of NSA, and which is responsible for encouraging the federal collection, distribution, application, and guidelines of trusted automated information systems.

4. Network security shall employ high speed cryptography while maintaining a minimum of 160 MB/s (1280 Mb/s) network bandwidth in each direction, i.e. 320 MB/s, 2560 Mb/s @ full duplex. Analytical validation of the crypto module's ability to sustain this throughput shall be provided.

5. For the Phase 1 design, and for Phase 1 demonstrations, keys shall be manually inserted into a look up table which is accessed by the cryptographic function.

4.3. HARDWARE DESIGN

The secure router and its included cryptographic module is the central hardware building block of the SHARE architecture. The secure router incorporates the physical and data link protocol layers for interfacing the two associated SANs. The crypto module performs the functions of encryption, decryption, authentication, key agility, key management, and alarm detection and response.

4.3.1. Objectives

The basic hardware objective of the SHARE•HPSC network is to provide a scalable MLS design which provides high bandwidth, low latency performance for heterogeneous inter–SAN operation. To this end, the secure router is designed as an intrinsic SHARE component. Cooperating routers are used to establish a secure network between separated SANs over a physically insecure public network. SHARE is envisioned as a connectionless–oriented network in that error tolerance and error recovery features shall be accommodated by higher order protocols.

The design and placement (location) of the secure router with its cryptographic functions shall include consideration of the following competing features:

1. Maximize the ability and flexibility to provide various levels of security as needed by any given network implementation. This includes being compatible with the system scalability approach.
2. Minimize the hardware size, weight, and box cost. This is to include consideration of a hardware migration path which goes from 'box to module to multichip–module (MCM)'.
3. Maximize the expectation of National Security Agency (NSA) certification. This includes minimizing and containing software code and working with NSA to ensure the certification of the SHARE security aspects in general, and of the SHARE secure router and crypto module in particular.

4.3.2. The Secure Router Design Approach

A hardware design shall be provided as a scalable secure implementation to support the SHARE environment. There is no requirement to produce hardware in Phase 1.

The cryptographic module resides in a box which is connected between the local SAN and the Myrinet switch via physical, data link, and network protocols as shown in Figure 2.

The following lists the basic Phase 1 router requirements:

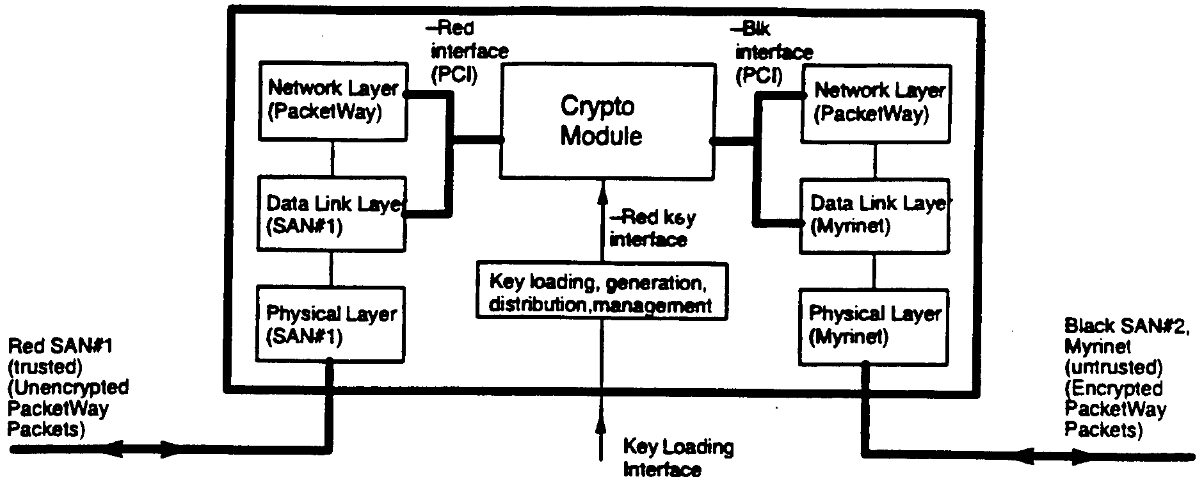

–Red interface (PCI)
–Blk interface (PCI)
Network Layer (PacketWay)
Crypto Module
Network Layer (PacketWay)
Data Link Laye (SAN#1)
–Red key interface
Data Link Laye (Myrinet)
Key loading, generation, distribution,management
Physical Layer (SAN#1)
Physical Layer (Myrinet)
Red SAN#1 (trusted) (Unencrypted PacketWay Packets)
Black SAN#2, Myrinet (untrusted) (Encrypted PacketWay Packets)
Key Loading Interface

Figure 2: Secure Router Approach

1. The block diagram design of the secure router shall be completed by Phase 1 program end. This diagram shall illustrate where hardware/software COTS and other standardized elements are anticipated, and where new design is required.
2. At the router level, with all network multilevel security and data integrity features implemented, demonstrate that throughput projections meet 160 MB/s, and 320 MB/s full duplex. At this time the router shall provide:
   –compatibility with the commercial Myrinet HPC LAN
   –compatibility with the SHARE node to network architecture
   –Security provisions:
   - A scalable security architecture which includes a cryptographic module
   - A security architecture which supports MLS protection. For Phase 1, single level protection shall be implemented.
   - Routing paths and host connections which consider subject authorizations (clearances)
3. Compatibility of the design with the Myrinet interface for (1) 10 bit single–ended 1 meter protocol and (2) 9 bit differential 25 meter protocol shall be provided.
4. In Phase 1, the secure router architecture and design shall work with the Myrinet LANai chip for the SHARE demonstration.
5. The functionality of the secure router shall be simulated in the Phase 1 program end demonstration. The message passing latency vs router placement trades shall be indicated. Analysis and simulation shall project the performance of the secure router which includes the cryptographic module.
6. Goal: A shrink migration path of the cryptographic module to 1/2 of a 6U VME form board (including secure memory allocation) shall be shown.
7. Capability of the design to support multilevel security, i.e. support key agility on a per–message basis and allow the assignment of algorithms and/or keys based on sensitivity level.

8. The provided Phase 1 hardware supporting documentation shall include:
   - router interface descriptions
   - router box layout
   - router box pinout
   - number of boards and their functionality
   - IC type and quantity (estimate summary per board)
   - other EEE components (estimate per board), i.e. electrical, electronic, and electromechanical items

A PCI (Peripheral Component Interconnect) data interface (or other open system standard interface providing the required bandwidth) to the crypto module shall be used to provide the high bandwidths required. Two PCI bus are perceived, a black bus for the untrusted public network side, and a red bus for the local SAN side. Connections to the crypto module are shown from both the data link and crypto layers in order to transmit/receive the plain text and the security context information required to process a packet.

The required key loading interface is also shown in the figure. The keys may be symmetric session keys, network private and public keys, and/or Message Authentication Code (MAC) and digital signature keys as determined during the course of the Phase 1 development. The design of the secure router shall contain provisions for the loading, generation, distribution, and management of the keys required to support the network security policy.

Figures 3 and 4 provide illustrations of the hardware processing and message composition concepts envisioned for message processing and formatting. It is anticipated that the final implementation will likely be a variation on these concepts. Figure 3 illustrates message processing that occurs at each functional element. Figure 4 is purposely the same figure, but with representative message formats illustrated.

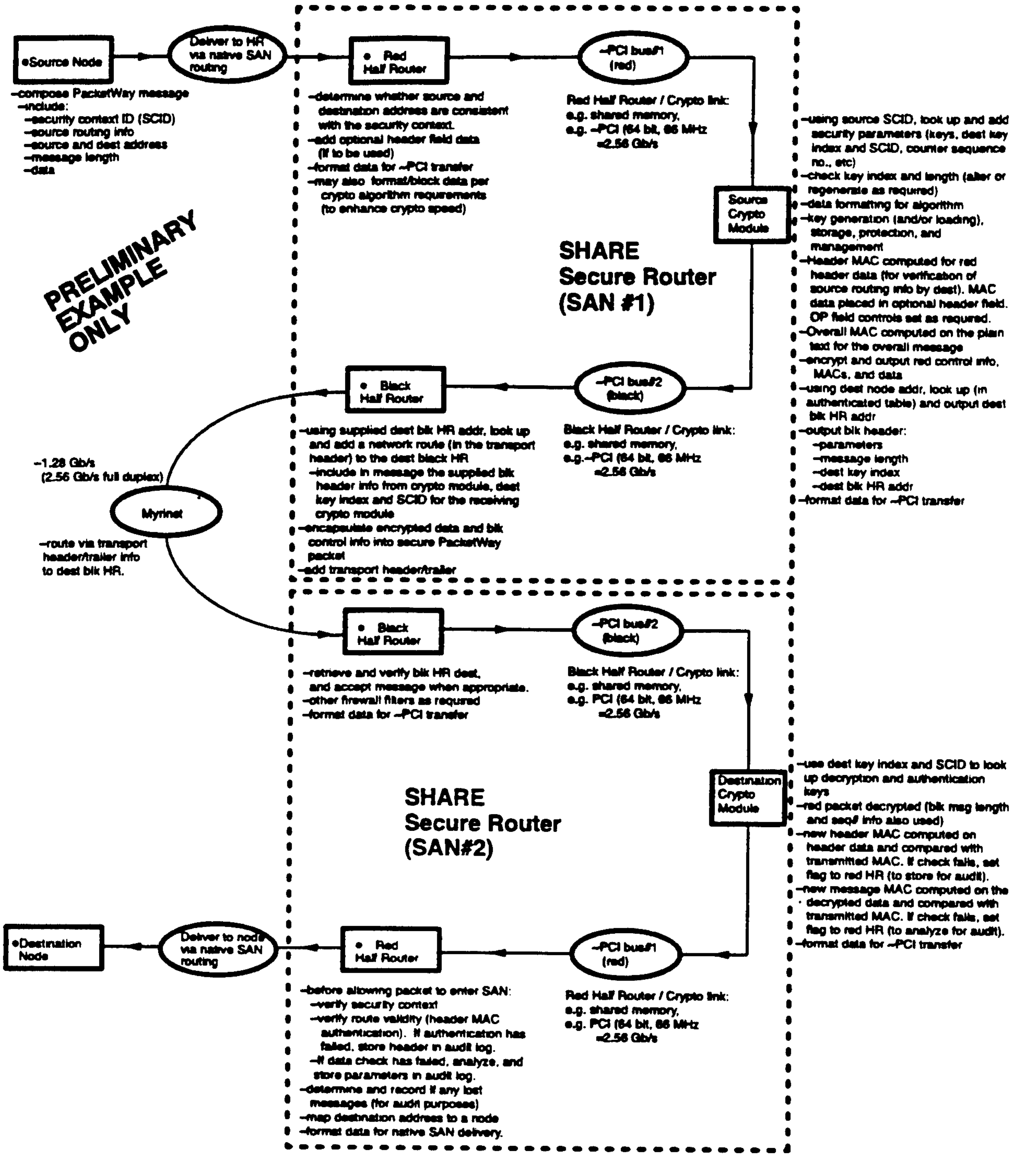

•Source Node
Deliver to HR via native SAN routing
–compose PacketWay message
–include:
–security context ID (SCID)
–source routing info
–source and dest address
–message length
–data
PRELIMINARY EXAMPLE ONLY
• Red Half Router
–determine whether source and destination address are consistent with the security context.
–add optional header field data (if to be used)
–format data for ~PCI transfer
–may also format/block data per crypto algorithm requirements (to enhance crypto speed)
~PCI bus#1 (red)
Red Half Router / Crypto link:
e.g. shared memory,
e.g. ~PCI (64 bit, 66 MHz
≈2.56 Gb/s
Source Crypto Module
–using source SCID, look up and add security parameters (keys, dest key index and SCID, counter sequence no., etc)
–check key index and length (alter or regenerate as required)
–data formatting for algorithm
–key generation (and/or loading), storage, protection, and management
–Header MAC computed for red header data (for verification of source routing info by dest). MAC data placed in optional header field. OP field controls set as required.
–Overall MAC computed on the plain text for the overall message
–encrypt and output red control info, MACs, and data
–using dest node addr, look up (in authenticated table) and output dest blk HR addr
–output blk header:
–parameters
–message length
–dest key index
–dest blk HR addr
–format data for ~PCI transfer
SHARE
Secure Router
(SAN #1)
~PCI bus#2 (black)
• Black Half Router
Black Half Router / Crypto link:
e.g. shared memory,
e.g.~PCI (64 bit, 66 MHz
≈2.56 Gb/s
–using supplied dest blk HR addr, look up and add a network route (in the transport header) to the dest black HR
–include in message the supplied blk header info from crypto module, dest key index and SCID for the receiving crypto module
–encapsulate encrypted data and blk control info into secure PacketWay packet
–add transport header/trailer
~1.28 Gb/s
(2.56 Gb/s full duplex)
Myrinet
–route via transport header/trailer info to dest blk HR.
• Black Half Router
~PCI bus#2 (black)
–retrieve and verify blk HR dest, and accept message when appropriate.
–other firewall filters as required
–format data for ~PCI transfer
Black Half Router / Crypto link:
e.g. shared memory,
e.g. PCI (64 bit, 66 MHz
≈2.56 Gb/s
Destination Crypto Module
–use dest key index and SCID to look up decryption and authentication keys
–red packet decrypted (blk msg length and seq# info also used)
–new header MAC computed on header data and compared with transmitted MAC. If check fails, set flag to red HR (to store for audit).
–new message MAC computed on the decrypted data and compared with transmitted MAC. If check fails, set flag to red HR (to analyze for audit).
–format data for ~PCI transfer
SHARE
Secure Router
(SAN#2)
•Destination Node
Deliver to node via native SAN routing
• Red Half Router
~PCI bus#1 (red)
–before allowing packet to enter SAN:
–verify security context
–verify route validity (header MAC authentication). If authentication has failed, store header in audit log.
–If data check has failed, analyze, and store parameters in audit log.
–determine and record if any lost messages (for audit purposes)
–map destination address to a node
–format data for native SAN delivery.
Red Half Router / Crypto link:
e.g. shared memory,
e.g. PCI (64 bit, 66 MHz
≈2.56 Gb/s

Figure 3: Message Processing Through the SHARE Network

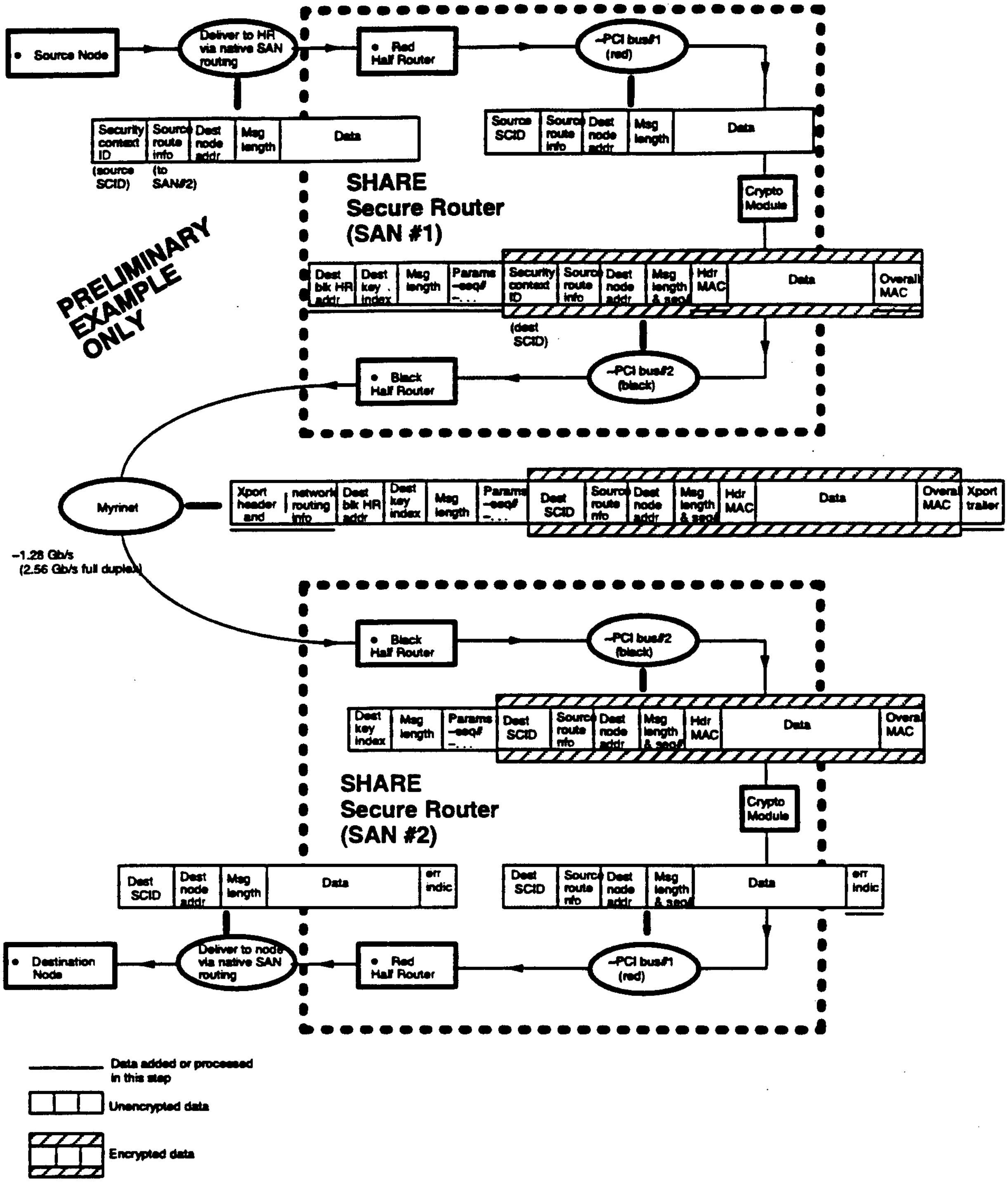


Figure 4: Message Composition Through the SHARE Network

4.4. SOFTWARE DESIGN

Phase 1 shall develop a system level software architecture design, and provide related research, analyses, and tradeoffs leading to this design. A system software architecture model shall be presented as a result of the analyses.

The following general features shall be incorporated into the software architecture:

1. Secure PacketWay shall be used as the network protocol. Software module development shall be cognizant of the ongoing work being performed on the protocol draft standard.
2. Message Passing Interface (MPI) protocols and libraries shall be used by the application. Included in these libraries shall be the secure functions and drivers necessary to support multilevel security. The using equipments shall take into account the ongoing work being performed on the protocol draft standard to include Secure MPI.
3. Software prototype modules, elements, and functions shall be written in C/C++ utilizing best commercial standards and practices.

Software shall be delivered to support the Phase 1 development. The software shall include functional elements representing:

1. PacketWay security extensions (i.e. per proposed draft standard)
2. MPI security drivers and libraries
3. Trusted network real–time operating system
4. Security policy software
5. Myrinet Control Program having:
   - Device driver modifications
   - Instrumentation features
6. Demonstration software having:
   - Benchmark performance tests
   - Other performance tests
   - Display / evaluation of results
   - Drivers for external instrumentation hardware
   - Other prototype implementation fragments
7. Software design elements including:
   - Interface descriptions
   - Design algorithms
   - Architecture models

Any additional specific Phase 1 software development beyond these items shall be confined to the development of that software required to support the Phase 1 demonstrations. Where possible, this software shall be written in portable modules that may be used in the Final Implementation described in section 5. Before developing specific Phase 1 modules, implementers shall refer to the more general requirements delineated section 5. to determine if there is a Phase 1 module partitioning that maximizes portability to the Final Implementation.

4.5. SYSTEM TESTABILITY

The on–line and off–line system testability requirements for the Final Implementation are defined in Section 5.4. No System Testability features are required to be implemented in Phase 1, however, the

Phase 1 design shall accommodate the System Testability features planned for the Final Implementation as much as possible.

4.6. COTS and Cost–Risk

Commercial Off–The–Shelf (COTS) products and existing technologies shall be employed in the Phase 1 design and demonstrations where possible. This approach is in recognition of current government and military initiatives to thereby improve cost, risk, and procurement time factors. COTS usage will also promote the insertion of the developing SHARE technology into future commercial applications.

4.7. DEMONSTRATIONS

There shall be three Phase 1 demonstrations. These demonstrations shall provide SHARE design feedback and show technical cumulative progress. The cryptographic demonstration elements shall build on previous results and conclude with a low latency key switch (goal = 0.7us) for key tables with a large number of user authorization entries (goal = 65,000). The three demonstrations shall include:

1. Demo #1: A preliminary demonstration is recommended to examine any known SHARE throughput and implementation issues, and to extract parameters for use in subsequent simulations. Existing hardware and software, cryptographic and otherwise, shall be used to the maximum extent possible.
2. Demo #2: A testbed shall be constructed and used to emulate and evaluate critical elements within the SHARE network. An emulation of the crypto module shall be included. Performance parameters determined from Demo #1 shall be used where appropriate. The testbed shall examine basic SHARE security policy tenets via the ability to send secure packets (using Secure PacketWay) over a homogeneous network. The testbed construction shall not preclude its eventual migration to MLS. Secure network connectivity demonstration shall be provided via testbed connections to various node types including PCs, SUNs, the HPC environment, the improved MCP (Myrinet interface), and potential units having other operating systems.
3. Demo #3: A simulation of the full SHARE network, with emulated crypto, shall be provided for performance benchmarking and system analysis. The simulation shall include examination of the effects associated with a very large scale SHARE network implementation, e.g. the resulting performance effect of 100s of network nodes on critical designs such as the SHARE dynamic routing algorithm. Realistic performance characteristics shall be used as derived from Demo #2.

5. SHARE•HPSC FINAL IMPLEMENTATION PLAN

5.1. SYSTEM / SECURITY DESIGN

In this section, the Phase 1 System/Security Design described in section 4.2. is extended to the planned Final Implementation. There is necessarily some repeat of the Phase 1 requirements here. Also, refer to section 4.1., "Summary: Phase 1 Outputs", item 1, for a description of the 'Final Implementation' architectural design output anticipated from Phase 1.

5.1.1. System Architectural Requirements

The goals of the SHARE•HPSC system design shall be those of a high performance, connectionless oriented, secure packet switching network:

1. Performance: The primary operating parameters that shall be maximized are high bandwidth, low latency, and reliable operation in an error environment.
2. Network throughput is of primary concern and shall take into account:
   - Algorithm(s) execution and complexity
   - Communication link parameters, the number of allowable hops, and routing strategy
   - Trusted and untrusted path delays, including firewalls, including 'busy path' routing alternatives
   - Hardware and software latency
   - Hardware and software modularity 'penalties'
   - Scalability 'penalties' incurred by the extendable network
   - Multiple security levels
3. A Network Manager shall be provided that monitors, coordinates and controls the SHARE environment and, with the NOS, ensures its integrity. Its responsibilities shall include but are not limited to:
   1. –Secure access control of the Network Manager itself.
   2. –Authentication of the local SANs.
   3. –Verification of the security and access control functions of the SHARE routers.
   4. –Network key management.
   5. –Coordinate security services.
   6. –Ensure network security integrity, e.g. stop the transmission of classified information when a security hazard occurs.
   7. –Disaster recovery.
   8. –Provide console for the display and control of network operation.
4. The Network Manager shall reside in a trusted location (a secure room) whose security level is equal to that of the highest level SAN in the SHARE network.
5. SAN independence: There shall be nothing in the SHARE•HPSC design that precludes its connection to a SAN of any type from any vendor.
6. Transparent addition of new networks and services: SHARE shall permit the connection of additional SANs and additional services with, ideally, zero interruption to the current

users of the SHARE environment. The ability to add additional SANs shall be essentially limitless.

7. On–line verification and off–line testability: SHARE shall incorporate an on–line monitor function that continuously verifies the integrity of the cryptographic elements in particular, and the environment in general. SHARE shall also continuously certify the network's ability to transfer authentic data (see item 8).
8. End to end testing: This protocol shall provide confirmation to a source host of the expected integrity of the received data at the destination. This will enable the secure crypto routers to establish a virtual channel and to output data before completing receipt of the input data. The router is thus able to maintain high throughput rates without requiring large memory capacity. Each end to end (E to E) test shall be accomplished with a minimum of packets so as to minimize network loading. E to E retests shall be performed when 'non acknowledge' replies and retransmit requests are received at the source.
9. Tolerance to subnetwork failures: The failure of a particular network path or subnetwork shall not affect the operation of the overall SHARE environment and shall not limit the ability of the network to transmit/receive secure data from/to subnetworks which remain operational.
10. Rapid and smooth failure recovery: SHARE shall provide for recovery from the failures identified via items 7. through 9. above, and from other more generalized failures, e.g. loss of some packets in a message, with minimal impact to users.
11. Guidelines: The design of the SHARE system and Secure PacketWay shall use the TCP/IP protocols as guidelines for the services that are to be provided. However, it is to be understood that to achieve the high speed, low latency performance required by SHARE cooperating SANs, some streamlining and reduction of TCP/IP services which unduly restrict performance is expected.

5.1.2. Specific Security Requirements

SAN authentication ensures that no covert communication, storage, or timing channel can exist which might otherwise compromise secure dialog among the SHARE participants. A node on the same public network which is not authenticated, is not able to converse with a SHARE participant, and SHARE nodes can not respond in to unauthenticated nodes.

The following specific TCSEC security aspects shall be implemented in SHARE:

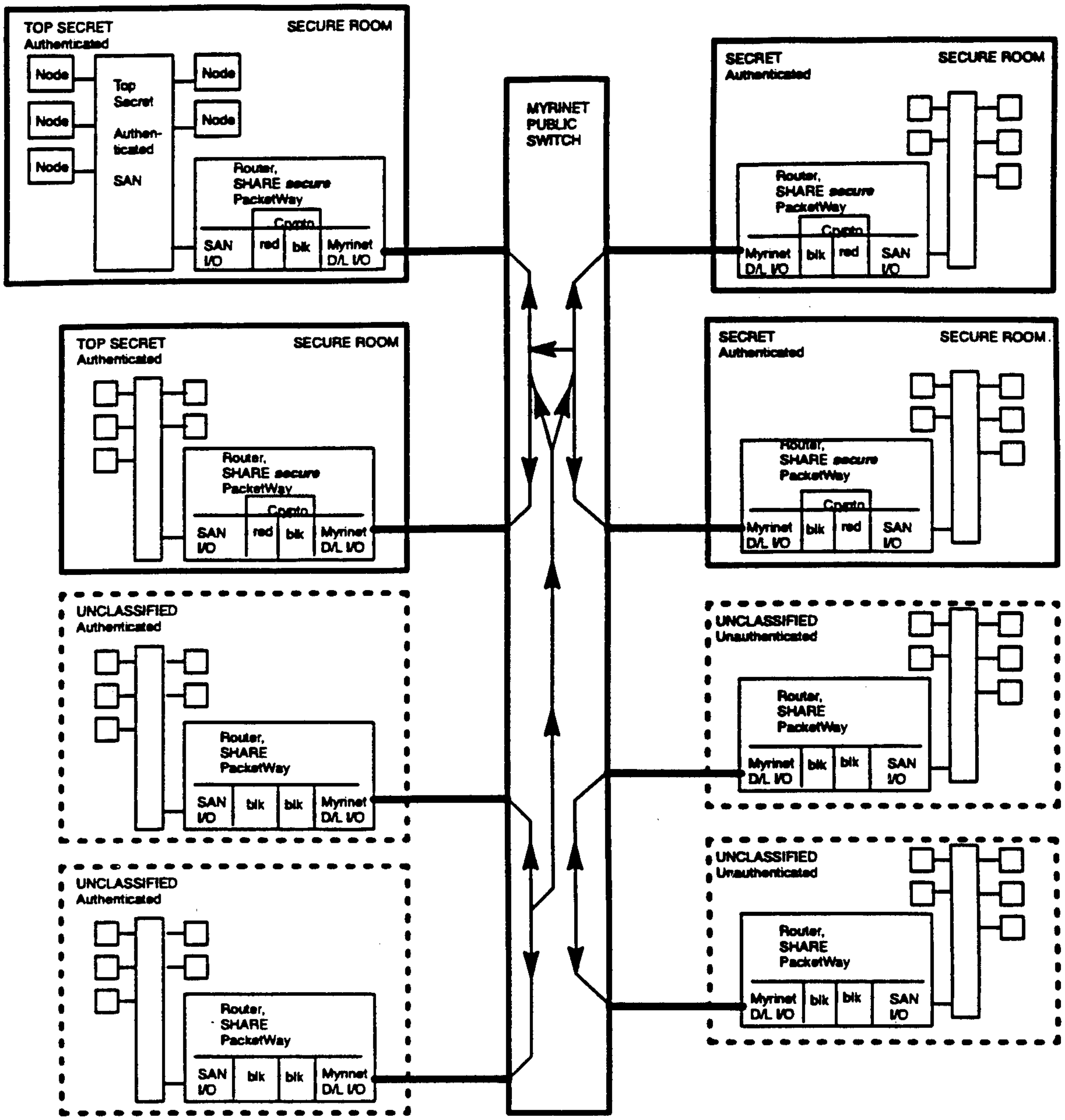
1. Multilevel security protection shall be provided for the SHARE environment. Each local SAN shall be interfaced to the SHARE network by a secure router (or equivalent) that manages the security of the data from/to the local SAN. Transmitted data from a SAN must be at the security classification of that SAN. Received data allowed to pass to a SAN shall be of that SAN's security level or lower. The secure routers shall be monitored and managed by the SHARE Network Manager and the Network Operating System.
2. Services provided by the trusted NOS shall include but are not limited to:
   –Managing the networking of the local SHARE routers.
   –Overseeing server/routers: users, backup, updates.
   –Organizing and implementing user access rights.
   –Providing login procedures.
   –Key generation and/or loading, protection, and distribution.

–Performing network monitoring and secure auditing services.
 –Performing breakin attempt monitoring.
 –Performing indication of abnormal traffic flow.
 –Tracking incorrect log ins and rescind login access.
–Providing self contained security features: passwords, password aging.
–Supporting application installation.
–File back up operations.
–Performing naming service, time stamp service.
–Have flexible and failsafe backup procedures for SHARE network components.

3. Security protection shall focus on data content rather than traffic analysis. A tutorial shall be developed listing the types of potential attacks and discussing the degree of protection provided for each.
   The secure router shall be effective against an attacker having:
   –the ability to view all of the bits traversing any of the untrusted links.
   –the ability to occasionally change (delete, replace, add, replay) any bits in any packet which traverse any of the untrusted links.
4. The secure transmission and reception of data by a host is to be accomplished within the framework of the following protocols:
   - Secure MPI (developed by MPI Forum MPI–2 working group,
   - Secure PacketWay (defined by the IETF PacketWay working group),
5. Network security shall employ high speed cryptography while maintaining a minimum of 160 MB/s (1280 Mb/s) network bandwidth in the transmit and receive directions (i.e. 320 MB/s, 2560 Mb/s @ full duplex).

Commercial and/or military community security cost/risk analysis shall be conducted. In addition to custom design, these analyses shall exploit the availability of COTS.

The example diagram in Figure 5 illustrates a connectivity example resulting from the effects of a particular MLS implementation.



NO OTHER COMMUNICATION PATHS PERMITTED between indicated SANs (per multilevel security protocol and per specific need to know authorizations).

Secure PacketWay routers have cryptographic modules and red/black partitioning.

Figure 5: An Example of Permitted Communication Paths in a SHARE PacketWay Network

5.2. HARDWARE DESIGN

5.2.1. Hardware Requirements, General

The cryptographic module design shall include the functions of encryption/decryption, key management, message authentication, alarm, alarm control, and alarm response. The alarms shall detect any event affecting, or potentially contributing to, encryption or key related compromise. The hardware and software security design shall not be compromised due to unauthorized access, tampering, conductive or radiated emissions, or other elements as appropriate for NSA certified systems and equipments.

The encryption/decryption implementation shall initially include consideration of the triple DES and IDEA algorithms. A migration path to Type 1 algorithms shall be shown. These and other algorithms may be implemented in different ways. Figure 6 depicts an encryption/decryption implementation using stored symmetric and MAC keys. In this implementation the XOR MAC employs the DES algorithm to generate the message digest. A DES pseudo random function is used to produce the one way hash used for authentication.

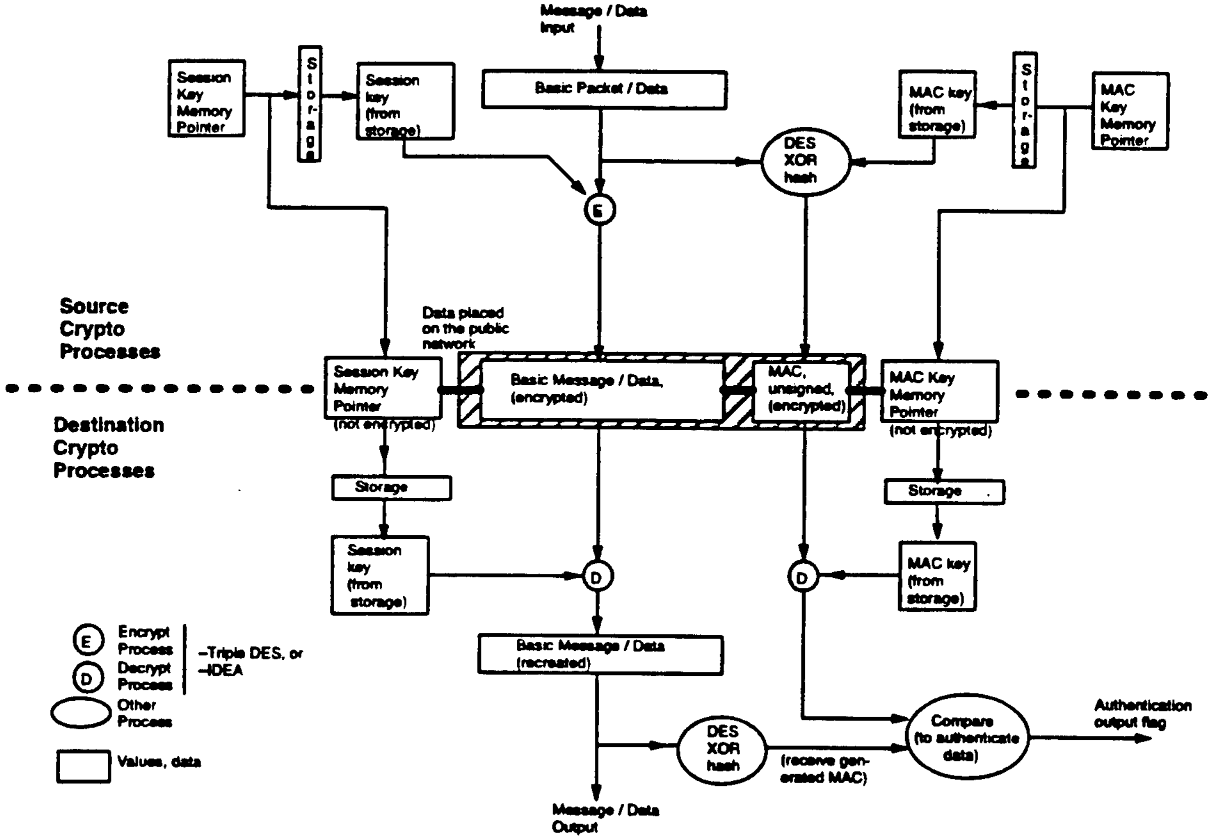

Message / Data Input
Session Key Memory Pointer
Storage
Session key (from storage)
Basic Packet / Data
MAC key (from storage)
Storage
MAC Key Memory Pointer
DES XOR hash
E
Source Crypto Processes
Data placed on the public network
Session Key Memory Pointer (not encrypted)
Basic Message / Data, (encrypted)
MAC, unsigned, (encrypted)
MAC Key Memory Pointer (not encrypted)
Destination Crypto Processes
Storage
Storage
Session key (from storage)
D
D
MAC key (from storage)
E Encrypt Process
D Decrypt Process
–Triple DES, or
–IDEA
Other Process
Values, data
Basic Message / Data (recreated)
DES XOR hash
(receive generated MAC)
Compare (to authenticate data)
Authentication output flag
Message / Data Output

Figure 6: Encryption/Decryption and MAC via Stored Symmetric Keys

A separate document is planned, "SHARE Cryptographic Key Generation–Distribution–Management". This document will detail the subject requirements. In general, methods for key generation–

distribution–management shall be traded off and an analysis provided which describes the rationale for the selected approach. The analyses shall cover key generation, distribution, key entry, agility, MLS, storage, usage, lifetime, backup, key output, (possible) escrow, archival, and destruction. The analyses shall consider the use of symmetric and public keys where appropriate. The "SHARE Cryptographic Key Generation–Distribution–Management Requirements" document is not yet available, and prior to its issue, the following abbreviated list of issues shall be addressed in the SHARE design. These issues/requirements will be repeated and amplified in the planned document.

1. Key generation:
   –Local (built–in) key generation,
   –Remote (separate box) key generation,
   –Quantity of symmetric (session) keys needed and their permissible lifetimes.
   –Quantity of public and private keys needed and their lifetimes.
   –Key (security level) partitioning.

2. Key Distribution and storage:
   Keys are used by the source to encrypt data and by the destination to decrypt data. In the general case, there may be advantages to source key distribution and destination key distribution by different means. For example, a symmetric session key may be generated by the source, used for encryption, and transmitted to the destination (with the data message) after being encrypted with the destination's public key. Figure 7 illustrates such a scheme, including provision for MAC and digital signature. This approach allows the secure routers to have autonomous symmetric key generation methods, i.e. it is not necessary for routers in the system to have an identical stored memory of symmetric key values. Also note that the public and private keys can be much longer than symmetric keys, can have greater lifetimes and less frequent loading intervals, since their use for encryption/decryption is applied against data of very short lengths. The SHARE development shall analyze and trade off factors associated with key distribution and storage including but not limited to:
   –Electronic vs manual key distribution.
   –Key loading method.
   –Off–line vs on–line (in the data message) key distribution.
   –Symmetric and/or public–private key usage.
   –Methods associated with assignment of session keys
   –Key distribution and storage safeguards.
   –Common vs autonomous router key storage, i.e. each router has a different key set (but key pairs must be compatible).

3. Key Management:
   –Physical security protection, other safeguards against electronic attack, and other forms of theft shall be addressed.
   –The Network Manager's role in key generation, distribution, and management.

At least three schemes for message authentication shall be examined, DES–XOR, MD5, and SHA.

Encryption and key failure alarms are discussed in System Testability, Section 5.4.

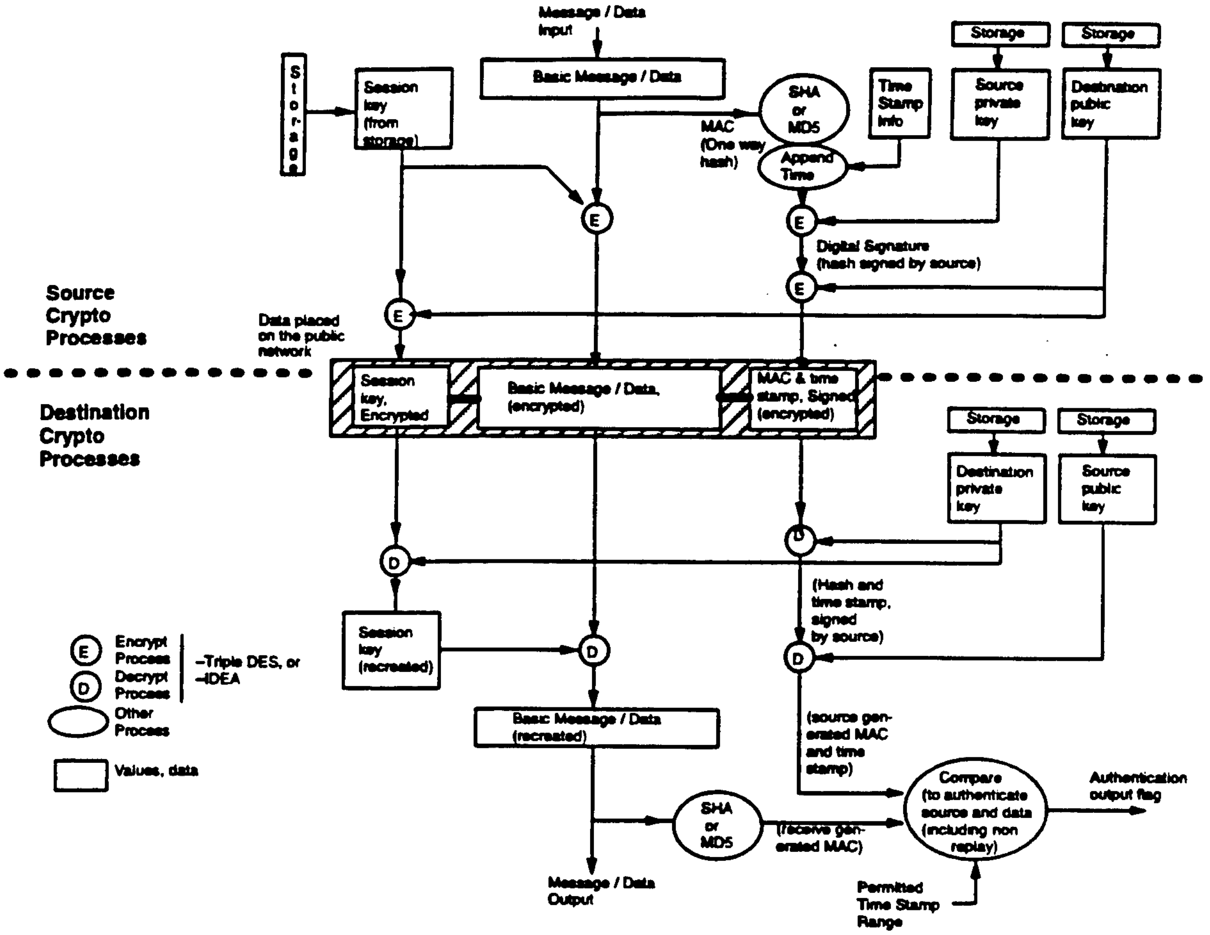


Figure 7: Encryption/Decryption via Real Time Symmetric Key Distribution,
and MAC, Digital Signature, and Time Stamp via Stored Public Keys

The illustration in Figure 7 is intentionally in the same form as Figure 6 to enable comparison of symmetric and public key methods. The figure depicts a crypto message processing implementation which includes the functions of encryption/decryption, real time session (symmetric) key distribution, message authentication, digital signature, and record/replay prevention. Note that only the intended recipient can verify that the message is errorless and authentic, and the message is non repudiatable.

The proposed SHARE environment shall incorporate the functions illustrated in Figures 6 and 7, although it is not mandated that they be implemented exactly as in the figures. Analysis and discussion of rationale is required to support the selected implementation, with specific comment on deviations from the functionality shown in the two figures.

5.2.2. Hardware Requirements, Specific

Additional specific requirements of the secure router implementation are:

1. The hardware shall be compatible with evolving industry standard network protocols, specifically Secure MPI and Secure PacketWay. It should also be compatible with the Sanders HPSC network of choice, Myrinet.

2. The secure router shall be able to properly process packets in both encrypted and clear format. A router without a cryptographic module shall be able to process clear packets and disregard encrypted packets.
3. The secure router shall provide a throughput of 160 MB/s (1.28 Gb/s) in each direction, i.e. a full duplex rate of 320 MB/s. This rate is compatible with a full speed Myrinet link. The latency (start–of–packet in–to–out delay) shall be minimized.
4. The hardware shall enable implementation of a B2 (TCSEC) level network ("Mandatory–Structured Protection", per the National Security Center 'Red Book', "Trusted Network Interpretation, NCSC–TG–005)" as a minimum. The extension to a Multi–Level Secure (MLS) network shall be provided.
5. The encryption/decryption capability shall include consideration of the triple DES and IDEA algorithms.
6. Red and black data, keys, and processes shall be partitioned into red and black isolated sections of memory such that it is not possible for the red to be accessed by the black.
7. Consideration shall be given to a crypto/router design that provides the ability to encrypt/decrypt with multiple algorithm and/or key options consistent with each local SAN's security level. The ability and method for upgrading to a programmable router/cryptographic module shall also be considered.
8. The functionality of the secure router shall be simulated in a demonstration.
9. The schematic design of the secure router shall be completed.

5.3. SOFTWARE DESIGN

5.3.1. Specific Software Requirements

This section presents some basic secure router concepts that influence its software design. Appendix A specifies the software development requirements.

Figure 8 is an abbreviated process control diagram for the envisioned secure router. The purpose of this diagram is to give insight into some of the major required software controls.

The local SAN and Myrinet interface blocks represent both the data link protocol and physical layer connectivity. A Red bus and a Black bus are represented as the anticipated method of achieving the high data speeds and security separation required to/from the crypto process (PCI bus type anticipated).

The formatting required by the crypto is shown within the crypto module since this data blocking must be compatible with the crypto algorithm in process. This algorithm may be selected by the security protocol of the source SAN and invoked dynamically in real time by PacketWay, or selected as a part of the particular Type 1 or Type 4 security equipment design incorporated into the secure router.

A rudimentary key loading process is illustrated, as is memory partitioning to separate red and black data.

Additional message processing details requiring software modules are illustrated in the block diagrams of Figures 3 and 4 of Section 4.3.2.

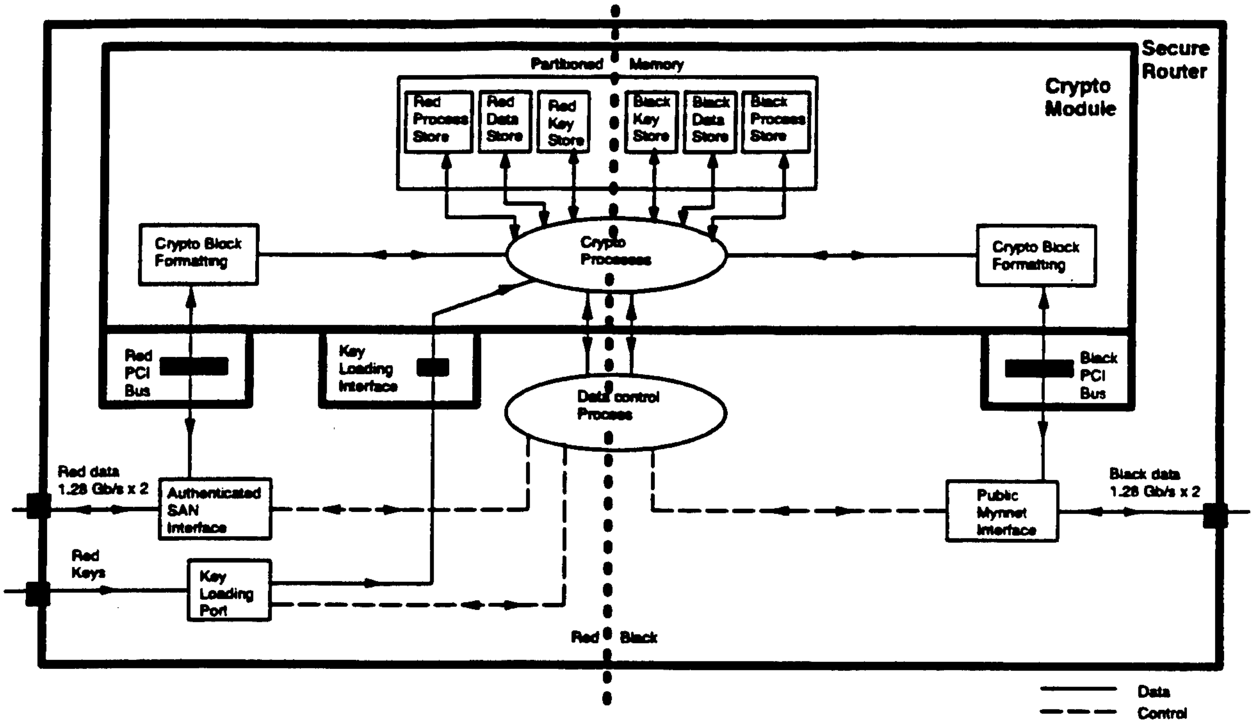


Figure 8: Process Control Diagram, Secure Router

5.4. SYSTEM TESTABILITY

The system monitoring and test functions are comprised of on–line and off–line tests as described below.

5.4.1. Hardware Testability

The SHARE system hardware at the box level shall have both an on–line and an off–line test capability. The on–line test shall continuously ensure that an equipment's cryptographic function is operating without security compromise, and that the network is expected to be able to transfer data when requested. The off–line test shall facilitate the isolation of problems, effect their timely repair, and return the system/equipment to operational status.

5.4.1.1. On–Line Hardware Testability

The hardware elements connected to the SHARE environment can be grouped as follows:

1. Hosts connected to the local SAN,
2. Secure Routers (or similar) connected between a local SAN and the public SAN,
3. Non Secure Routers (or similar) connected between an unauthenticated local SAN and the public SAN,
4. Other peripherals connected to the local SAN,
5. Public network components, e.g. Myrinet.

These hardware elements are associated with SANs of various security levels. Items 1 and 4 are located in a secure room with that room's security policy and test controls in effect. Items 3 and 5 represent unauthenticated items that are outside of the SHARE secure partition. The on–line testability of item 2, the Secure Router, is addressed below.

As part of the SHARE on–line test capability, each SHARE router (or equivalent) containing a cryptographic module shall provide a security watchdog function. This watchdog function shall consist of at least two parts: (1) an on–line continuous self monitoring function that verifies the security integrity of the cryptographic elements, and (2) an on–line inter router verification function that reports the probability of successful data transport between routers.

The self monitoring function shall trigger alarms, inhibit processing, and erase (zeroize) unencrypted keys when accepted thresholds are not met on key statistics, encryption statistics, decryption failures and failure statistics, interface activity or inactivity, power supply adequacy, box physical integrity, and exposure to adverse environmental conditions. Other monitors may be incorporated to further enhance security integrity and are expected to be added as the SHARE environment matures.

The inter router verification function shall consist of two automatically performed tests. A 'heart-beat' test shall be periodically performed between all SHARE authenticated routers which verifies the expected routability of transmissions when they are sent, and to inform system managers of an existing problem with the transmission links. The heartbeat test shall be of short duration and shall be performed by each router to each other authenticated router every TBD (60) seconds or less.

The second inter router verification test function shall be a more inclusive test and shall cause the initiation of randomly timed test data transfers. The test data transfers shall be round trip secure data transfers to/from other SHARE routers. Error free round trip data transfer shall be verified by the initiating router. The test data block shall consist of a minimum number of PacketWay packets so as not to adversely load the network. A successful error free round trip will enable the transfer of SHARE data to/from the verified router for a period of TBD (60) minutes. Successive test transfers shall be initiated randomly. The period from one test request to the next shall not exceed TBD (60) minutes.

When either the heartbeat or round trip inter router verification functions is not successful, host nodes shall be informed, and appropriate operator display made available. The host may then decide to initiate or delay data transfer.

5.4.1.2. Off–Line Hardware Testability

Each SHARE router (or other similar network component) shall provide an off–line self test capability that (1) enables the isolation of a problem to a plug in module, (2) verifies router performance before placing it on–line, and (3) performs the routines for placing itself on–line. Such test capability may be as simple as the substitution of a plug in card and a re–verification before on–line activation. This off–line re–verification may in fact use the on–line hardware if the on–line test function encompasses the necessary evaluation criteria, or it may be as complex as an independent test routine that provides more detailed specifics of operation (that are too time or memory intensive to incorporate into the on–line test).

Off–line test capability shall include implementation of all of the functions necessary to recover from minor faults, major faults, and even disasters. No external equipment shall be required to bring the SHARE environment on–line from any fault scenario. Off–line testing shall not reduce the network bandwidth or increase the latency of data transmissions to/from operating SANs.

5.4.2. Software Testability

Software modules shall be written so as to permit their individual execution and test before integration into the overall program. The on–line and off–line testability features defined in the above paragraphs shall be supported by independent software modules so as to facilitate their individual (hardware and software) test.

5.5. COTS and Cost–Risk

COTS products and existing technologies shall be employed in the Final Implementation design and demonstrations where possible. This approach is in recognition of current government and military initiatives to thereby improve cost, risk, and procurement time factors. COTS usage will also promote the insertion of the developing SHARE technology into future commercial applications.

The technical hardware design shall be supported by a cost–risk analysis.

6. DEFINITION OF TERMS

SHARE•HPSC: Secure Heterogeneous Application Runtime Environment for High Performance Scalable Computing

Security: In the SHARE environment, cryptographic methods and open standards are used to ensure information survivability in the transmission of (multi–level) secure packets of information over a public network. The network security objectives are data integrity, confidentiality and authentication, and availability.

Heterogeneity: SHARE is the hardware and software fabric that provides interoperation between inherently dissimilar network components, i.e. between SANs of various types and security levels, and between host nodes having differing bandwidth and operating system requirements.

High Performance: High performance here means high bandwidth, low latency, operation in an error environment, on–line and off–line testability, and controlled fault recovery.

Application Runtime Environment: The provided high speed, low latency network environment enables applications to run across cooperating heterogeneous SANs in near real time. Application programs use traditional Application Program Interface (API) software to access the underlying network services.

Scalable: Scalability is the ability to increase or decrease the size and heirarchy of a connected network of SANs using fundamental SHARE•HPSC building blocks. Very large scale extensions to the SHARE network of SANs are possible using basic building blocks which include, but are not limited to, secure and non secure routers, local SANs, and high speed public network switches.

. . . . . .

Application Program Interface – API: Software that can be referenced by an application program to access underlying network services, i.e. a set of software commands that an application may make to a software module providing a defined service, e.g. a cryptographic service.

Asynchronous Transfer Mode – ATM: ATM is a fast (25 to 622 Mbps defined), cell (small packet) based of method of moving voice, data, video, and other data and telecommunication oriented information from one location to another. ATM is connection oriented with asynchronous cell reception. The cell consists of a 5 byte header with routing and other network related information, and a 48 byte information field for data, images, voice, and/or video. Although the 3 layer architecture, i.e. Physical, ATM, and ATM Adaptation Layer, does not map to the OSI framework directly, ATM comprises lower level protocols that map most closely to the OSI physical and data link layers.

Audit: A function performed by a computer system (or network) which monitors and records selected system activities so that actions affecting security can be traced to the responsible party. Audit information must also be protected. Examples of events that are included in network audits are the establishing and dropping of a network connection, the occurrence of lost, misrouted, or timed out data, valid and invalid log in attempts, and the failure of a network component.

Authenticate: To establish the validity of a claimed identity, e.g. data, sender.

Automated Information System – AIS: The assembly of computer hardware, firmware and software configured to collect, communicate, control, create, disseminate, manipulate, process, and store data and/or information. (DoD 5200.28).

Backbone: A high capacity network that links together other networks of lower capacity. The highest level segment in a heirarchical network, i.e. the main segment in a structured wiring approach. Example: A backbone may inter connect different buildings across a distributed plant.

Bridge: In LANs, a device that employs the bottom two OSI layers to interconnect dissimilar networks. Bridges are used to segment networks based on network traffic, e.g. by putting traffic from workstations that frequently communicate on a common LAN. This is done by filtering algorithms internal to the bridge which operate on the source and/or destination addresses, or other criteria.

Certificate Authority: A trusted central repository of public key information. The information needed to enforce the security policies of the cryptographic system may be provided by this authority or, in more rigorous implementations, in other servers dedicated to access control.

Certification: The technical evaluation of a system's security features, made as part of and in support of the approval/accreditation process that establishes the extent to which a particular system's design and implementation meet a set of specified security requirements.

Commercial–Off–The–Shelf – COTS: Specifies standard commercially available hardware or software that can be purchased 'immediately' and without modification.

Communications Security – COMSEC: Protective measures taken to deny unauthorized information access derived through communications, and to employ a security policy to authenticate communication. (NCSC–WA–001–85).

Computer Systems Laboratory – CSL: See National Institute of Standards and Technology.

Connectionless Network: A class of communication service in which no connection between source and destination is established prior to data transmission. It is analogous to mailing a letter. This type of network delivers the packets individually, in random order, and without guarantee of correctness nor of delivery (some may be lost). A connectionless network has no network wide acknowledgements, flow control, or error recovery, although these services may be provided by higher level protocols and/or on a link–by–link basis.

Connection Oriented Network: A class of communication service that establishes a virtual circuit before transmission of data, i.e. the destination address is contacted and a path defined through the network prior to transmitting data. It is analogous to making a telephone call. A connection oriented network provides a number of data integrity services not provided by a connectionless network (and therefore requires more overhead), e.g. flow control, error checking, error recovery, etc. This network type delivers the packets in sequence and confirms the delivery. Depending on the quality of service, the delivery may be guaranteed error free.

Context: In an MLS system, a security context is the particular set of security definitions associated with the message, e.g. "a session key is assigned to each message based on its security context."

Covert Channel: A communication channel that allows two or more collaborating processes to transfer information in a manner that violates the system's security policy. Also related are Covert Storage Channel, and Covert Timing Channel.

Covert Storage Channel: A covert channel that involves the direct or indirect writing of a storage location by one process and the direct or indirect reading of the storage location by another process. Covert storage channels typically involve a finite resource (e.g. sectors on a disk) that is shared by two subjects at different security levels.

Covert Timing Channel: A covert channel in which one process signals information to another by modulating its own use of system resources, e.g. computer time, in such a way that this manipulation affects the real response time observed by the second process.

Data Encryption Standard – DES: A symmetric key algorithm (uses the same key for encryption and decryption). Defined in FIPS PUB 46–2.

Datagram: Generally, the unit of data (with appended header) that is passed from the network layer to the data link layer. The datagram packet is transmitted over a connectionless network and contains sufficient information to be routed from source to destination independently of previous transmissions, i.e. a self contained packet, independent of other packets.

Degree of Protection: Computer data can be protected to varying degrees based on a multilevel security model and multiple subject authorization keys. SHARE provides for 65000 subject authorizations.

Discretionary Access Control: The means of restricting access to objects based on the identity of subjects and/or groups to which they belong. The controls are discretionary in the sense that a subject with certain access permission is capable of passing that (perhaps indirectly) on to any other subject (unless restrained by 'mandatory access control').

EEE: The generalized classification of all electrical componentry, i.e. Electrical, Electronic, and Electromechanical.

Firewall: A firewall protects a given network, e.g. LAN, from unauthorized users and/or untrusted networks from which intrusions can originate. It is the chief instrument used to implement an organization's network security policy. All data sent to or from users outside of this network must pass through a firewall computer, or set of computers, that checks, routes, and labels all information that passes through it. A firewall generally operates at the highest layers of the OSI model, i.e. from the Network to the Application layers, and may perform processing at all seven layers. Also see Screening Router and Trusted Firewall.

Flow Control: A function performed by a receiving entity to limit the amount or rate of data sent by a transmitting entity – a technique for assuring that the transmitting entity does not overwhelm the receiving entity with data.

Gateway: A device that interconnects dissimilar LANs that employ different high level protocols. A gateway encompasses all 7 layers of the OSI reference model.

Host: Any computer based system connected to the network and containing the necessary protocol interpreter software to initiate network access and carry out information exchange across the communications network. Example: A terminal is not a host because it does not contain the protocol software needed to perform information exchange; a workstation (by definition) is a host because it does have such capability.

Hub: A multiport hub, or repeater, connects network nodes or terminals and performs signal amplitude and timing restoration. It takes the incoming bit stream and repeats it to all other ports (except the originating port). The hub is the center of a star topology, i.e. one node, one port. Intelligent hubs also contain network management, access control, and fault diagnosis features.

International Traffic in Arms Regulations – ITAR: Regulations covering the export of arms, including cryptographic methods.

Internet Engineering Task Force – IETF: a subgroup of the Internet Activities Board. IETF supports the advancement of technologies through the cooperative work of representatives of the industrial forces in the computing industry. Information is disseminated, documents are published, and standards developed in the many working groups which are intended to give some direction to Internet development activities.

Key Management: The generation, storage, distribution, deletion, archiving, and application of keys in accordance with a security policy. Key management includes ensuring that key values generated have the necessary properties, making keys known in advance to the parties that will use them. and ensuring that keys are protected as necessary against disclosure and/or substitution. All keys have limited lifetimes.

Local Area Network – LAN: A network, typically in the Mbps range, wherein all segments of the transmission are situated in an office, building, or campus environment. Ownership is by the user organization.

Lockheed Martin Communication Systems – LMCS: A SHARE•HPSC subcontractor.

Mandatory Access Control: A means of restricting access to objects based on the sensitivity (as represented by a label) of the information contained in the objects and the formal authorization (clearance) of the subjects to access information of such sensitivity.

MCNC: A SHARE•HPSC subcontractor.

Message Passing Interface – MPI: The host/SAN interface protocol used in SHARE•HPSC. Secure MPI is an extension that contains the security provisions utilized by SHARE•HPSC.

Mississippi State University – MSU: A SHARE•HPSC subcontractor.

Multilevel Device: A device that is used in a manner that permits it to simultaneously process data of two or more security levels without risk of compromise. To accomplish this, sensitivity labels are normally stored on the same physical medium and in the same form as the data being processed, i.e., machine–readable or human–readable.

Multilevel Secure – MLS: A class of system that provides a capability for various levels and categories or compartments of data to be stored and processed in an AIS, and permits selective access to such material concurrently by users who have differing security clearances and need–to–know. The identification, segregation, and control of users and sensitive material on the basis of security clearance, material classification category, and need–to–know are essentially under automated control. Also see TCSEC.

Myrinet: The high speed network and switch to be used in the early demonstrations of SHARE performance. Myrinet is capable of 1.28 Gb/s (160 MB/s) data transmission, and 2.56 Gb/s full duplex transmit/receive.

Myrinet Control Program – MCP: The operating program residing in the LANai integrated circuit which interfaces the Myrinet LAN, i.e. one particular interface implementation.

National Computer Security Center – NCSC: Part of the National Security Agency, NCSC is responsible for encouraging the federal collection, distribution, application, and guidelines of trusted automated information systems. NCSC is the government agency responsible for evaluating trusted systems, i.e. systems developed to the requirements in the 'Orange' and 'Red' books. NCSC works closely with NIST, Defense Intelligence Agency (DIA), and other government agencies with an interest in computer security.

National Institute of Standards and Technology – NIST: In 1987, Congress passed the Computer Security Act, which authorized NIST to develop standards for ensuring the security of sensitive but unclassified information in government computer systems. NIST's Computer Systems Laboratory (CSL) sets standards for computer security, conducts research, tests security products, and provides security training and support to other agencies. Official standards are produced as FIPS, Federal Information Processing Standards.

Network Architecture: The conceptual description of the way communication is accomplished between data processing equipments at different sites. It also specifies the processors and terminals, protocols, and software that must be used.

Network Interface Unit – NIU: A general term for any network interface adapter which is placed between a node and a network cable.

Network Trusted Computing Base – NTCB: The totality of protection mechanisms within a network, the combination of which is responsible for enforcing the network's security policy. The hardware, firmware, and software composing a network system that is responsible for enforcing a security policy.

Node: The point from/to which directed data originates/culminates, i.e. whatever can send and receive packets.

Object: A passive entity that contains or receives information, e.g. records, blocks, pages, segments, files, directories, directory trees and programs, as well as bits, bytes, words, fields, processors, video displays, keyboards, clocks, printers, network nodes, etc.

Open Systems Interconnection – OSI: A framework for network architectures consisting of 7 layers, e.g. physical, data link, network, transport, session, presentation, and application. The bottom three layers support the components of the network necessary to transmit a message, the next three layers generally pertain to the characteristics of the communicating end systems, and the top layer supports the end users.

"Orange Book": Document DOD 5200.28–STD, "Trusted Computer System Evaluation Criteria". Also see TCSEC.

Packet assembler / disassembler – PAD: A PAD can be defined as hardware or software that performs the function of putting data not conforming to a particular packet specification into a compliant packet, e.g. a PAD may put non X.25 data into X.25 packets.

PacketWay: PacketWay, previously known as MessageWay, is the network layer protocol used in SHARE•HPSC. Secure PacketWay is an extension that contains the security provisions utilized by SHARE•HPSC.

Peripheral Component Interconnect – PCI: An advanced high speed data bus, i.e. defined as 32/64 bit, 33/66 MHz, 132 to 528 MB/s.

Port: A number which is generally identified with a process running on a host, i.e. which application program is to receive the incoming traffic (allows multiple user programs to communicate concurrently with the one application program). A port is usually a small uP with its own separate clock, memory, registers, and often, a CPU (a full fledged micro computer).

Protocol Control Information: A header. Info exchanged by peer entities (layers) at different sites on the network to instruct an entity to perform a service function. e.g. sequence check field, error check field, compress field.

Protocol Data Unit – PDU: SDU + Protocol Control Information: User data + header (from application layer). Information that is delivered as a unit between peer entities of a network.

"Red Book": Document NCSC–TG–005, Version 1, "Trusted Network Interpretation". Also see TCSEC.

Repeater: Repeaters are used to extend the cable length or the number of workstations allowed per segment, i.e. if the cable length or the number of attachments exceeds the value that the network cable can support, a repeater can be installed between two segments. Also see hub.

Router: A device that employs the bottom three OSI layers to interconnect remote and/or dissimilar networks. The router segments network traffic based on the routing algorithm, the destination network layer address, a higher layer protocol, and/or the associated LAN facility in use.

Sanders: A Lockheed Martin Company. The prime SHARE•HPSC contractor.

Screening router: A screening router performs packet filtering functions at the I/O interfaces of the router and rejects network traffic not meeting the filter rules. It is generally defined to operate at the Transport and Network layers of the OSI model, e.g. at the TCP/IP layers. TCP and IP headers of incoming and outgoing packets are examined and the packets rejected or passed based on the packet filter rules. Since the router operates at the lower OSI layers, its operation tends to be more transparent than a firewall. A screening router may be used as the first line of defense in the creation of a firewall.

Secure Router: A router which interconnects trusted hosts on their own subnetwork with external untrusted systems. The secure router provides security services, e.g. cryptographic functions, for the trusted hosts when they communicate via external untrusted systems.

Security Policy: The set of laws, rules, and practices that constrain and control security relevant activities in the management, protection, and distribution of sensitive information.

Service Data Unit – SDU: User data transferred transparently by layer N+1 to layer N, and subsequently to N–1.

Socket: For TCP/IP, socket = port# + network address of the host that supports the service, e.g. socket = port# + IP address, where IP address = Network Address + Host Address.

Source Routing: In source routing, the routing logic (a list of addresses of intermediate nodes) is inserted into the data before the data is transmitted, e.g. by the source workstation. This relieves the intermediate nodes of having to store and update complex routing tables, which in turn gives the nodes more latitude to perform other necessary functions, e.g. network management functions. When the source routing list has been completed, the destination address field is used for routing to the final destination. Note: Directory routing uses a directory at each node defining the preferred and possibly the second preferred outgoing link for each destination.

Subject: An active entity, generally in the form of a person, process, or device, that (1) causes information to flow among objects or, (2) changes the system state.

Subject Security Level: A subject's security level is equal to the security level of the objects to which it has both read and write access. A subject's security level must be dominated by the clearance of the user the subject is associated with.

System Area Network – SAN: SANs are packet networks made of point to point links with flow control and using source routes. Modern SANs have high throughput, low latency, and low error rate, and are usually confined within a room, building, or plant. A SAN is generally an 'independent island' incapable of direct intercommunications with other SANs and high performance LANs.

System Security Officer – SSO: The person responsible for the security of a system. The SSO is authorized to act in the "security administrator" role. The SSO role includes such items as specification, change, and maintenance of subject and object security characteristics, trusted system configuration, and auditing.

Switch: Switches are similar to bridges, but typically have more ports. Each of the ports may be dedicated to a segment or device, e.g. a server. Transmitting devices are allocated the full network bandwidth on demand. Switch interfaces may be provided with different transmission rates.

Trusted: The belief that a system meets its specifications, and in data security, pertaining to hardware and software systems that have been designed and verified to avoid compromising, corrupting, or denying sensitive information.

Trusted Computer Security Evaluation Criteria – TCSEC: A pair of documents developed by NCSC, i.e. DoD 5200.28–STD, the "Orange" book, and NCSC–TG–005, the "Red" book, setting a policy standard of a basic set of requirements which defines (evaluates) degrees of assurance in an AIS. The TCB and NTCB of a SHARE system are currently defined to be Class B2, "Mandatory–Structured Protection". B2 systems require discretionary and mandatory control for all subjects and objects, covert channels are addressed, the TCB must be carefully structured into protection critical and non critical elements, the TCB interface must be well defined and enabled to be subjected to thorough testing and review, strong authentication mechanisms are provided, trusted facility management is provided, and stringent configuration management controls are imposed. The "Red" book, "Trusted Network Interpretation", provides interpretations of the "Orange" book for trusted computer/communications network systems. The "Red" book thus extends the TCSEC "Orange" book to networks of computers and describes a number of additional security services that arise in conjunction with networks.

Trusted Computer System: A system that employs sufficient hardware and software integrity measures to allow its use for processing simultaneously a range of sensitive or classified information.

Trusted Computing Base – TCB: The totality of protection mechanisms within a computer system, including hardware, firmware, and software, the combination of which is responsible for enforcing the system's security policy.

Trusted Firewall: A formal and deliberate separation of information, processes, and applications by a TCB AIS using a security policy and/or ancillary devices to establish communication with not classified but sensitive processes.

Trusted Path: A mechanism by which a trusted person at a terminal can communicate directly with the TCB. This mechanism can only be activated by the person or the TCB and cannot be imitated by untrusted software.

Trusted Subnetwork: A subnetwork containing hosts and routers that trust each other not to engage in active or passive attacks and trust that the underlying communications channel, e.g. Myrinet, isn't being attacked.

Type 1: A classified cryptographic algorithm or device used in defense applications and approved by NSA for protecting classified information.

Type 2: A classified cryptographic algorithm or device that is approved by NSA for protecting sensitive unclassified information in systems involving intelligence, national security, and certain military activities.

Type 3: A Type 3 cryptographic algorithm or device is one that is approved as a Federal Information Processing Standard (FIPS) and is used for protecting sensitive unclassified information.

Type 4: Type 4 is a commercial cryptographic algorithm or device that is not based on Federal Information Processing Standards (FIPS). NIST will maintain a Computer Security Objects Register (CSOR) listing these items.

Virtual Circuit: See Connection Oriented Network. A connection oriented network establishes a virtual circuit before the transmission of data, i.e. the destination address is contacted and a path defined through the network prior to transmitting data. All packets follow the same route, need not carry a complete address, and arrive in sequence. It is analogous to making a telephone call.

X.25: CCITT Recommendation X.25 (X=data communications study group, 25=study group number) was first drafted in 1974 and is now updated every 4 years. It specifies the interface between (1) user equipment and (2) an X.25 packet switched, connection oriented network.

802.X: IEEE committee 802 is divided into subgroups which are devoted to a series of lower–layer network protocols called the 802.X series. Project 802 was named for the date of its inception, i.e.

1980, 2nd month (February). The 802 subgroups range from those supporting established protocols, to those devoted to the advancement of already implemented protocols, to those supporting protocols by corporations and other organizations, to those incorporating protocols from other entities. Two example subgroups are (1) 802.3 'carrier sense multiple access with collision detection' (CSMA/CD), is similar to Ethernet, will operate with Ethernet, and uses the same basic technology, and (2) 802.6 incorporates the ANSI X3T9.5 fiber standard which defines FDDI.

7. REFERENCES

SHARE DOCUMENTS:

1. "SHARE•HPSC Network Security Architecture, Rev–1", Sanders, a Lockheed Martin Company, 13 June 1996, Document # PUBS–96–C30–W.

TECHNICAL DOCUMENTS FROM VARIOUS SOURCES:

2. "DoD Trusted Computer Evaluation Criteria", CSC–STD–001–83, 1983, DoD 5200.28–STD, Library No. S225,711, National Computer Security Center, 1985, "Orange Book".
3. "Trusted Network Interpretation, of the Trusted Computer Evaluation Criteria", NCSC–TG–005, 31 July 1987, "Red Book".
4. "Technical Report Evaluation Criteria for Cryptography", (Temporary Draft), Common Criteria for Information Technology Security Evaluation, V 0.99b, 3/30/96.
5. "Guide to Understanding Trusted Facility Management", NCSC–TG–015, Library No. S–231,429, National Computer Security Center.
6. "Design of a Key Agile Cryptographic System for OC–12c Rate ATM", Stevenson, Hillery, Byrd, Gong, and Winkelstein, MCNC Internet Society Symposium on Network and Distributed System Security, IEEE Computer Society Press, 1995.
7. "Proposed Specification for the MessageWay Protocol", IETF Internet Draft, Danny Cohen; Myricom, Craig Lund; Mercury Computers, December 14, 1995.
8. "Principles of Key Management", Fumy & Landrock, IEEE Journal on Selected Areas in Communications, Vol 11, No. 5, June 1993.
9. "A Platform for Heterogeneous Interconnection Network Management", Warrier and Sunshine, IEEE Journal on Selected Areas in Communications, Vol 8, No. 1, Jan 1990.

LOCKHEED MARTIN DOCUMENTS:

10. "Software Engineering Methodology Handbook", V 4.0, Lockheed Martin, Jan 17, 1992

BOOKS:

11. "Applied Cryptography", Schneier, John Wiley & Sons, Inc, second edition, 1996.
12. "Computer Networks", Black, PTR Prentice–Hall, Inc, second edition, 1993.
13. "The McGraw–Hill Internetworking Handbook", Taylor, McGraw–Hill, 1994
14. "Computer Security Basics", Russell and Gangemi, O'Reilly & Associates Inc, 1991
15. "Guide to Security and Data Integrity (LAN Times)", Farley, Stearns, Hsu, Osborne McGraw Hill, 1996.
16. "Computer Communications Security", Ford, Prentice Hall PTR, 1994.
17. "Information Security", Longley, Shain, Caelli, M Stockton Press, 1992
18. "Internetworking", 2nd edition, Miller, M&T Books, 1995

8. APPENDIX A – SOFTWARE DEVELOPMENT REQUIREMENTS

8.1. FINAL IMPLEMENTATION

In the Final Implementation Phase, significant new software development shall employ an accepted design methodology which organizes the development and minimizes technical risk. Significant new software development is defined to be an object consisting of TBD (20) lines of C/C++ code, or a collection of objects consisting of TBD (200) lines of C/C++ code. The design methodology shall include the use of a life cycle model and object oriented design. It is recognized that a complete conversion of existing practice to object oriented design may not be realistic, and that elements of structured analysis and design may also be used.

SHARE software shall undergo rigorous planning, design, review and examination, testing, and 'break in attempts' to ensure that software security threat activities will not be successful.

The following paragraphs summarize the general requirements associated with software development.

8.1.1. Life Cycle Model

Significant new software development shall include the use of a life cycle model, e.g. waterfall, evolutionary spiral (preferred), or other model to assist in planning and tracking the development. The model shall cover the period from Requirements Definition through Software Test and Integration. The purpose of the model is to provide a 'time ordered' process in which all activities, milestones, and deliverables map into program plans and schedules. It is believed that this will assist in reducing technical risk and uncertainty and in increasing manageability.

8.1.2. Object Oriented Design

In order to promote modularization, portability and design reuse, significant new software development shall be based on an object–oriented approach. In this approach, identifiable design phases shall include:

1. Analysis Phase:

   – establish the system requirements and partition the system; build the object, state, and process models; validate the model; and restate/refine the requirements.

2. Design Phase:

   – design architectural components; choose the implementation flow of control; allocate system components of processors and tasks; handle boundary conditions; address external events; design application components; update object, state and process model; validate, iterate, and refine the analysis model; refine the system requirements to reflect the results of this design phase.

3. Implementation Phase:

   – code and test the Computer Software Units (CSUs): analyze and check inputs; plan CSU code and test; specify unit test cases; code CSUs; conduct code walkthroughs; run CSU test and document results; conduct CSU test results walkthroughs; update requirements baseline; identify reusable components.

4. Testing and Validation Phase:

   – integration testing of classes; testing inheritance structures; testing aggregation structures.

8.1.3. Structured Analysis and Design Phases

Significant new software development may be subjected to structured analysis and design. The following five design phases shall be recognized in the software's development.

1. In the Software Requirements Phase, the following activities shall be performed:

   – establish and understand the system requirements; partition the system; build the software requirements model (define the top level functional requirements, the interface requirements, the control behavior, and the performance requirements); and document the results of this phase as the software requirements baseline. In this phase, Data Flow Diagrams (DFDs) and a Data Dictionary (DD) shall be constructed.

2. In the Preliminary Design Phase, the following activities shall be performed:

   – develop the software architecture; allocate functional, interface, and performance requirements to software components; develop the software test approach; and document the results of the preliminary design. In this phase, the DFDs and DD are updated, Flattened Flow Diagrams (FFDs) and Task Communication Graphs (TCGs) are created from FFDs, Timing Diagrams and Task Process Diagrams are generated from TCGs, and a First Cut Design is generated from Task Process Diagrams.

3. In the Software Detailed Design Phase, the following activities shall be performed:

   – finalize the software design including all processing and interface definitions; allocate the requirements to each detailed design component; develop the software test procedures and test cases; and document the results of the detailed design. In this phase, structure charts and information models may be created.

4. In the Software Coding and Unit Testing Phase, the following activities shall be performed:

   – build the source code; debug the source code; conduct unit level testing; and, finalize software build plan and procedures for Computer Software Code (CSC) testing. In this phase, the following elements are checked and verified: coding standards and compliance and C/C++ language usage checks; syntax and semantics checks; code coverage and optimization checks; run time checking for bounds errors and memory leaks; application workstation performance analysis; measures of wall and execution time by component for all code executed, including COTS; evaluations per Halstead and McCabe metrics.

5. In the CSC Integration and Testing Phase:

   – activities associated with the integration and test of the software components in the system shall be performed. In this phase, software complexity and test coverage metrics may again be employed.

6. In the Computer Software Critical Item (CSCI) Integration and Testing Phase:

   – *formal* integration and test of the software components shall be performed.

8.1.4. Integrated Software Development Environment

The use of an integrated software development environment is encouraged to promote uniformity of design.

SANDERS
A Lockheed Martin Company

SHARE*HPSC
Network Security Architecture

Rev - 1

12 June 1996

This document was developed under the Secure Heterogeneous Application Runtime Environment for High Performance Scalable Computing (SHARE*HPSC) program.

Document # PUBS-96-C30-W

SHARE*HPSC Network Security Architecture

Rev - 1

12 June 1996

Approved by:	For:	Date:
Jeff Smith	Sanders	6/13/96
Greg Byrd	MCNC	6/12/96
Robert George	MSU	6/13/96
Jack Brizek	Lockheed Martin Comm. Systems	5/31/96

Overview

This document describes the proposed network security architecture for SHARE*HPSC, a Secure Heterogeneous Application Runtime Environment for High Performance Scalable Computing, hereafter abbreviated as SHARE. The goal of the SHARE research is to develop a high-bandwidth secure network and runtime environment to support heterogeneous, distributed, scalable computing.

The SHARE network, shown in Figure 1, connects smaller high-performance networks known as SANs (System Area Networks). A SAN can range from a single workstation or personal computer (PC) to an embedded special-purpose scalable multiprocessor.

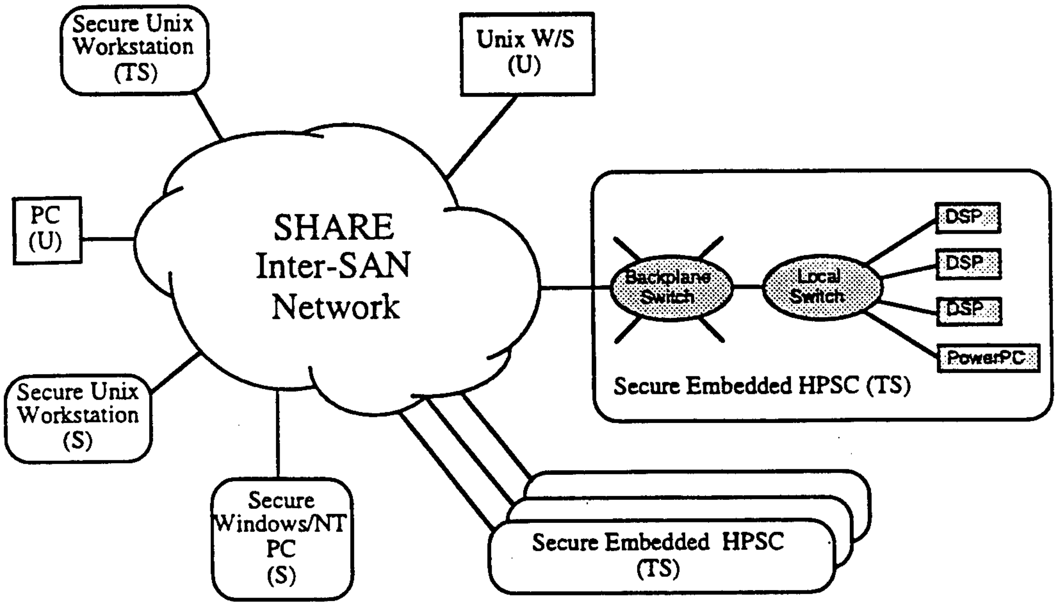


Figure 1 Example of SHARE environment.

SHARE is designed to allow secure SANs to communicate over a non-secure network. Each SAN operates at a given security level, such as Top Secret (TS), Secret (S), or Unclassified (U). SANs may communicate with other SANs at the same or lower level, depending on the network security policy.

SHARE provides hardware and software support for the secure delivery of messages between SANs. This document describes the components that make up the SHARE security architecture, including high-speed cryptographic hardware and security extensions to network and application layer software.

Network Security

For the purposes of this document, network security is defined as the set of services used to assure the secure delivery of data between secure SANs. Secure delivery means that data is not delivered to unauthorized parties and that data may not be altered during transmission between the source and the intended destination. Security at this level will be accomplished through cryptographic functions and protocols. (See Appendix A.)

The assumed threat on the inter-SAN network is an active adversary which has physical control of one or more network links. Any data that traverses compromised links may be read by the adversary, and data may be arbitrarily inserted, removed, or replaced on the link. It is the purpose of the security architecture to insure that an attacker cannot read messages sent from one secure SAN to another or create a message that will be accepted as genuine by a secure SAN, either by forgery or by replaying a previous legitimate message.

It is not the goal of SHARE to provide protection of the network resources themselves or to guard against all types of attacks. In particular, here are some issues that are not within the scope of the project:

- SHARE specifically does not include protection of resources within a SAN. Each SAN is assumed to be physically secure, consistent with its security level. For each node connected to a SAN, the operating system is presumed to provide strong authentication of users and protection of processes and data from the actions of other processes. The network protocol software is also assumed to be trusted and will not allow data to be accessed by unauthorized processes.
- SHARE does not protect the inter-SAN network against denial-of-service attacks, in which an adversary effectively disrupts the ability of the network to deliver data between SANs. Complete protection against denial-of-service attacks is beyond the scope of this effort. For example, the attacker may physically disable a link or destroy all packets that cross a link; these issues should be addressed by designing the network to tolerate a reasonable number of faulty or disabled links. Network security protocols, however, should protect against more subtle denial-of-service attacks, if possible, and should avoid creating new opportunities for attack.
- SHARE provides limited protection against traffic analysis. The original message header, including source and destination addresses and message type, will be hidden through encryption. However, the attacker may be able to identify the source and destination SAN. SHARE does not manufacture or randomize messages to obscure traffic patterns. If such measures are necessary, they must be provided by the application or some other higher-layer protocol.
- SHARE does not ensure the survivability of the inter-SAN network or guarantee the operation of the physical links.

SHARE Infrastructure

SHARE uses high-performance standard networking components to support its goal of high-performance heterogeneous distributed computing.

High-performance distributed computing requires a high-bandwidth, low-latency interconnect between hosts. While the SHARE environment is independent of the specific interconnect technology, the Myrinet [Boden95] local area network was selected for the development of a proof-of-concept system. Myrinet bidirectional links operate at 160 MB/s in each direction. Links are connected in arbitrary topologies through multi-port switches[1], which support wormhole routing [Dally87] for low latency. A specialized host interface chip, known as the LANai, is used to support streamlined data transfers between the network and the host memory. Myrinet supports messages of arbitrary length[2], avoiding the need for segmentation and reassembly.

The advantages of a high-performance interconnect can easily be squandered through inefficient or poorly implemented network protocol software. SHARE will be built on top of the MessageWay protocol [Cohen96], a proposed inter-SAN protocol which is designed to support low-latency

[1] Four- and eight-port switches are now available, with 16- and 32-port switches in development.

[2] In practice, the length is limited by the implementation of the Myrinet Control Program (MCP). The point here is that data is not "packetized" for transmission over the network.

communication. MessageWay is a network layer protocol; it fills a role similar to that of the Internet Protocol (IP). MessageWay, however, is different from IP in several important ways:

- a smaller, dynamically assigned address space, designed to allow closely located hosts to be collected into a single MessageWay environment,
- no error detection (i.e., no per-packet checksum), though errors that are reported from other layers may be acted on,
- support for wormhole routing, in which a routing decision can be made as soon as the message header arrives, and
- the ability to use SAN-specific source routes for low-latency delivery of messages.

It is expected that SHARE will utilize MessageWay communications directly (through either an API or a communications library), but applications built for IP may also be supported by encapsulating IP packets inside MessageWay messages.

Finally, it is important that applications written for a heterogeneous environment be portable across a range of system environments. In order to provide a common communications framework, SHARE applications will be based on MPI, the Message Passing Interface [MPI94]. The MPI standard defines the user interface and functionality of a wide range of message passing capabilities. It has been ported to a significant number of machines, ranging from scalable multiprocessors to workstations.

As part of the SHARE research, extensions to MPI and MessageWay will be implemented to support high-performance secure message passing. These extensions will be offered to the appropriate standards bodies (IETF for MessageWay, MPI Forum for MPI), in the hopes that they be included in future versions of the standards.

Security Architecture

The following security functions have been identified as critical for SHARE:

1. *Encryption*. Private data transmitted between hosts must not be readable by unauthorized parties. Encryption and decryption should be supported at the data transmission rate; for this reason, symmetric key encryption will be used for data traffic. Encryption will also be used to support the secure distribution of keys; since high bandwidth is not required for key distribution, either public key or symmetric key algorithms may be used.

   Keys used for encryption may be changed on a per-message basis. Communication between different pairs of hosts may or may not use different keys. Multiple secure "sessions" may be in progress between a given pair of hosts; these may or may not use different keys.

2. *Authentication*. A message may be authenticated, so that the recipients may be assured of the origin and integrity of the message. Authentication should be supported at the data transmission rate; for this reason, an efficient message authentication code (MAC) should be used, rather than a digital signature based on public keys. Keys for authentication may be changed on a per-message basis.

3. *Key management*. Keys for encryption and authentication must be generated, distributed, and stored in a secure manner. Keys may be negotiated dynamically or loaded statically.

In order to support the above functionality, the security architecture consists of the following components, described in the remainder of the document:

- Hardware-based cryptographic module, supporting encryption, decryption, and authentication at the full Myrinet link rate.
- Extensions to MessageWay, known as Secure MessageWay, supporting encryption, decryption, and authentication of messages.
- Secure MessageWay router, which implements Secure MessageWay for inter-SAN messages. For SHARE, the secure router will utilize the cryptographic module to achieve high-bandwidth, low-latency, secure communication between SANs.

Figure 2 shows how these components are used to implement the SHARE security architecture. The cryptographic module is placed between the two half-routers (HRs) of the Secure MessageWay router. Messages on the secure ("red") SANs are not encrypted; Secure MessageWay is used to convey security information as needed between the nodes and the router for inter-SAN communication. The non-secure ("black") inter-SAN network uses regular MessageWay to deliver encrypted and authenticated messages between secure routers.

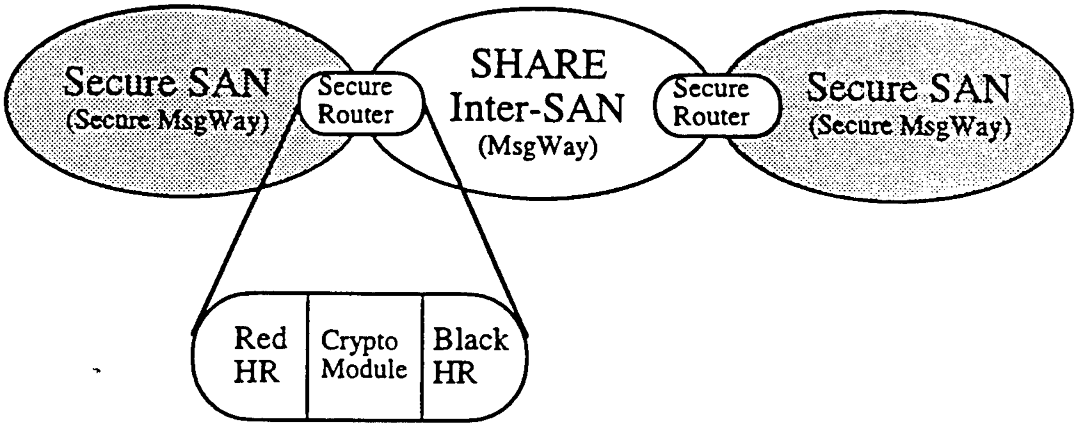


Figure 2. SHARE Security Architecture

Extensions to MPI, allowing security services to be visible to the application, are also anticipated and will be studied as part of the SHARE project. They will not be discussed further in this document, however, since the extensions deal with application-level concerns rather than with network security. Secure MPI will use the services provided by a secure network layer, such as Secure MessageWay.

Key management is a crucial part of the operation of any security environment. SHARE will support a variety of key management and distribution policies, ranging from statically loaded to dynamically negotiated keys. The details of key management for a specific environment are very dependent on security policies and other considerations. Specific key management strategies and implications for SHARE will be addressed in a separate document.

CRYPTOGRAPHIC MODULE

The cryptographic module is a hardware component designed to support the security architecture by providing encryption and authentication services at rates consistent with network bandwidth. Key agile operation will be supported, such that a different key may be associated with each message.

Requirements

The performance requirements for the SHARE cryptographic unit are as follows:

- Bandwidth: The cryptographic unit shall encrypt, decrypt, and authenticate data at a rate sufficient to match the Myrinet link rate -- 160 MB/s in each direction.
- Latency: There is no specific latency requirement for encrypting or authenticating data, but it is desirable to minimize the end-to-end message latency due to the cryptographic module.
- Key agility: The cryptographic module shall manage a large number (e.g., 2^{16}) of security contexts, each of which must contain one or more keys for encryption and/or authentication. The module must be able to switch keys quickly enough to ensure that each consecutive message may be encrypted using a separate key. The smallest possible Secure MessageWay message will be around 32 bytes, which implies a key switch latency of 200 nanoseconds or less for a 160 MB/s channel.
- Security: The cryptographic module shall be designed with an understanding of published security standards, enabling eventual certification (e.g., by the NSA) for government or military use.

Cryptographic Algorithms

In order for the cryptographic module to meet its performance requirements, it must utilize algorithms that support traffic rates of 160 MB/s (1280 Mb/s) in each direction. This high data rate implies a high-speed hardware implementation of a symmetric encryption algorithm.

In the following section, algorithms are identified which are suitable for the high data rates required by SHARE. For non-classified but sensitive data, DES is appropriate. For classified environments, high-performance Type 1 encryption devices should be used. The emphasis in the following sections is on DES, but the concepts are relevant to any high-performance symmetric encryption algorithm.

Data Encryption Algorithms

As discussed earlier, symmetric key algorithms are much faster than public key algorithms, so they are typically used for bulk data encryption. Current software implementations of symmetric algorithms, however, do not meet the bandwidth requirement, as shown in Table 1. Some algorithms, such as DES, are more suited to hardware implementation, because of bit-wise operations, but even software-efficient algorithms, such as Blowfish, are more than an order of magnitude too slow for this application.

Algorithm	*Bandwidth (MB/s)*
DES	1.1
IDEA	0.89
Blowfish	2.4
RC4	5.1

Table 1. Software performance of symmetric key algorithms (Pentium 120 MHz) [Dai96].

Since DES has been a federal standard since 1976, it is widely used and there are many commercial hardware implementations available. The Trusted Information Systems, Inc. survey of cryptography products [TIS96] lists 21 domestic sources of DES encryption chips. Most of these, however, still operate well below the rates required by high-performance networks

Digital Equipment Corporation has designed and fabricated a DES chip capable of encrypting at 1 Gb/s [Eberle93]. This chip is based on gallium arsenide (GaAs) gate array technology and has been made available on a limited basis. MCNC has used this chip in its research prototype for ATM cell encryption at OC-12c rates (622 Mb/s) [Stevenson95]. This is the fastest encryption device available for unclassified use. Even so, multiple chips will be required to meet the bandwidth requirements for SHARE.

One problem with using DES is its 56-bit key size; this is considered too small to provide adequate protection over reasonable periods of time. In 1993, it was estimated that a special key search engine could be built for $1 million which could find a DES key using brute force search in an average of 3.5 hours [Wiener93]. A $10 million investment reduces the search time to 21 minutes.

To improve security, triple DES may be used, in which two or three keys are applied to the data in an encrypt-decrypt-encrypt (EDE) pattern. Triple DES with two keys[3] is estimated to be a factor of 2^{52} times stronger than single DES.

Encryption Modes

DES and other block ciphers can be used in different modes to encrypt and decrypt data. The simplest mode is known as electronic codebook (ECB), in which each plaintext block is encrypted independently of every other block. Since a given plaintext always encrypts to the same ciphertext, an attacker can compare two encrypted messages for similarities, even if he cannot recover the plaintext. Insertions and deletions of data will also go undetected.

In cipher-block chaining (CBC) mode, each block of plaintext is XORed with the previous ciphertext block, prior to encryption. In this way, each ciphertext output depends on the previous outputs, and identical plaintext messages will encrypt into different ciphertext messages. An initial value (IV) must be provided, which is used to modify the first block of plaintext. A different IV must be used for each message; otherwise, messages which start with the same plaintext will also start with the same ciphertext.

Because CBC and similar modes introduce feedback into the encryption process, pipelining is not possible. The encryption of a block must be completely finished before the encryption of the next block may begin. The DES chip described above does not rely on pipelining to maintain its high throughput, so it is able to run at full rate for both ECB and CBC modes. If the same chip were used for triple DES, however, both latency and bandwidth would be reduced by a factor of three in CBC mode, because each block must wait for the previous result.[4] For ECB mode, the three DES operations could be pipelined across separate chips, thereby maintaining the full bandwidth.

To maintain high performance, encryption modes which use feedback must be avoided. Instead, we can use a technique known as long cycle chaining, in which a counter is XORed with the plaintext prior to encryption. This alters the plaintext, which avoids the problems associated with ECB, but does not require any ciphertext feedback. Identical plaintext messages will encrypt into different ciphertext messages, as long as a different initial value is used for the counter.

[3] One key is used for the two encrypt operations, while the second key is used for the decrypt operation in the middle. Using the same key for each operation yields the same result as single DES.

[4] This is the case if so-called "outer-CBC" is used, in which the output of the last encryption is used to modify the plaintext of the first encryption. An alternative is "inner-CBC," in which the feedback occurs for each of the three encryption/decryption steps. Inner-CBC is considered less secure [Schneier96, p. 361].

Military-Grade Encryption

In a classified military setting, the use of DES for encryption would not be allowed. High-security DOD applications require the use of Type 1 encryption devices.[5] High-speed Type 1 devices do exist; they are used in the KG-75 (FASTLANE) ATM encryption system and in the KG-189 SONET encryptor. These devices currently run at OC-12 rates (622 Mb/s) and would likely run at higher rates. Like the DES chips, however, multiple parallel devices would be necessary to achieve the bandwidth required by SHARE.

The devices and the algorithms that they implement are classified at the Secret level, so they would not be available for the current SHARE program. The SHARE hardware and software designs, however, should be easily adapted to a classified environment. The DES-based system that we propose here will be useful in non-classified environments and will serve as a valuable prototype for more secure environments.

Authentication Algorithms

Most well-known authentication algorithms are based on digital signatures. Because digital signatures generally rely on public key algorithms, however, they are too slow to implement full-rate authentication in SHARE. Instead, we will use message authentication codes (MACs), which are based on symmetric key algorithms.

A MAC that is gaining acceptance in the Internet community is known as keyed MD5 [Metzger95]. In this approach, a cryptographic hash function (MD5) is applied to the text to be authenticated and a shared secret key. For example, for message *m*, one could compute MD5(*k•p•m•k*), where *k* is a 128-bit key and *p* is 384 bits of random padding. (The '•' operator denotes concatenation.)

The fastest known software implementation of MD5 achieves 12.4 MB/s.[6] This is too slow for full-rate authentication, unless only a small part (e.g., the header) of a message requires authentication. Unfortunately, MD5 is not well suited to hardware acceleration, and we could only expect to get around 32 MB/s from custom CMOS hardware [Touch95]. .

An alternative which takes advantage of the fast DES chips described above is the DES-CBC MAC [FIPS113]. Data is encrypted using DES in cipher block chaining mode (CBC), and the MAC is the final output of the encryption. This MAC is only secure for fixed-size messages; otherwise, the MACs for messages A and B (using the same key) could easily be combined to form the MAC for A•B. Since SHARE message are expected to be variable in length, the DES-CBC MAC is not an appropriate choice.

The XOR MAC [BGR95] provides higher security than the DES-CBC MAC but still takes advantage of high-speed DES hardware. This MAC, illustrated in Figure 3, is based on XORing the output of a pseudo-random function with blocks of data from the message, then XORing all the partial results

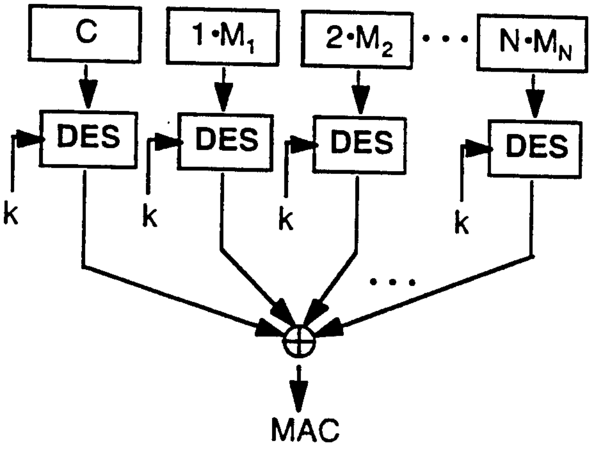


Figure 3. DES-based XOR MAC.

[5] Type 1 devices use classified encryption algorithms and are used to protect classified data. DES is a Type 3 algorithm, an unclassified algorithm that published as a FIPS standard and is used to protect unclassified sensitive or commercial information.

[6] DEC 4100/710 with 190-MHz Alpha processor [Touch95].

together to get the final MAC. An initial counter is included, as well as sequence numbers with each data block. DES can be used as the basis for the pseudo-random function. We divide the message into 32-bit blocks and prepend each block with a sequence number. Without the sequence number, a reordering of the message data would result in the same MAC as the original message. The sequence numbers also prevent the concatenation of messages described above.

Because of the use of sequence numbers, the MAC requires roughly twice as many DES encryptions as the DES-CBC MAC. However, all partial results can be computed in parallel, and the XORs may be computed in any order, so multiple DES chips can be used in parallel to speed up the calculation.

Summary

Because the cryptographic module designed for SHARE is a research prototype, it is expected that the security provided by single DES is sufficient as a proof of concept. For a higher level of security, the design may easily be modified to support triple-DES. A production version of the crypto module will likely require a custom ASIC or a government-supplied encryption device, both for performance and for improved security.

Any packaging assumptions that are made for the prototype design would also be subject to change in a final product. The crypto module could be designed as a separate box, or as a board or multi-chip module integrated with an embedded system. The prototype design may be used to estimate the feasibility and cost of these various approaches.

Because of high bandwidth requirements, the crypto module designed for SHARE will use the GaAs DES chip developed by Digital Equipment Corp. This chip is not really COTS, since it is not a commercial product, but it can be made available in small quantities for prototypes.

Two DES chips will operate in parallel to support the full 1.28 GB/s unidirectional bandwidth required for SHARE. Four chips will be required to compute the XOR MAC at the required rate. A total of 12 DES chips, therefore, is required for the SHARE cryptographic module: two for encryption, two for decryption, four for authentication, and four for verification. Long cycle chaining will be supported, but CBC will not.

Secure MessageWay

SHARE proposes to implement security functions as part of the MessageWay protocol for the following reasons:

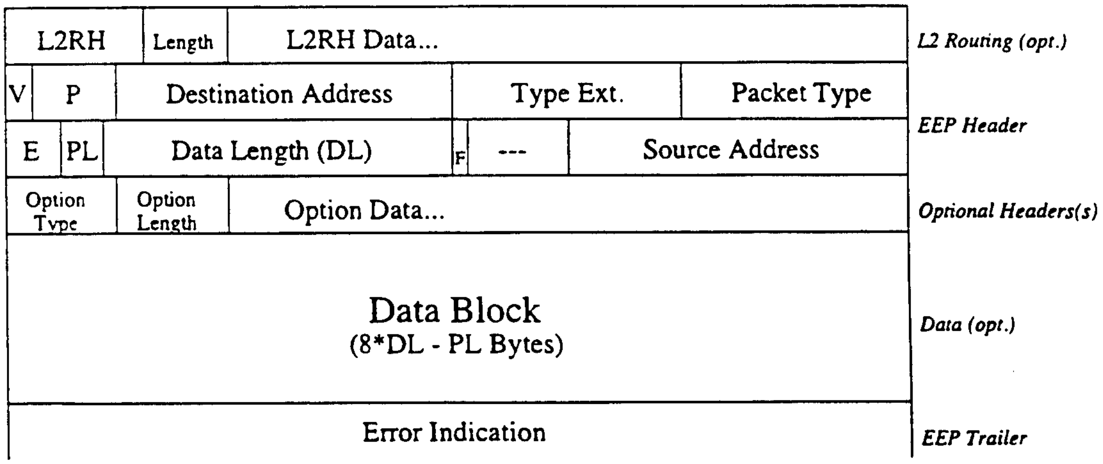
- MessageWay is the lowest layer in the protocol stack that is common across the heterogeneous collection of platforms envisioned for SHARE. Placing security at a low layer allows all higher layers to take advantage of the services. Moving the services to a lower layer would require a different solution for each network type.
- MessageWay is the lowest layer at which end-to-end communication is supported. If security were provided at the link layer, end-to-end encryption and authentication would be difficult or impossible to achieve. End-to-end services may also be utilized by applications.
- The contents of the MessageWay header are vulnerable to attack and should be protected. Changing fields in the header could cause the packet to be misrouted or misinterpreted, both of which are more serious than mere denial-of-service attacks.

If security is only provided at the MessageWay layer, then link layer protocols are left unprotected. For some environments, exposure of the link layer framing and flow control information may be an unacceptable risk. In a Myrinet network, for instance, an attacker who controls a link could disrupt service by sending an FRES (Forward Reset) symbol on the link [Myricom96]. In cases where local link control can have a global impact, additional security features, such as bulk link encryption, may be warranted. These link layer services are independent of the services provided by SHARE and are beyond the scope of the SHARE project.

MessageWay Packet Format

Figure 4 shows the format of a MessageWay packet, according to the March 1996 draft specification [Cohen96]. Source, destination, and type information are provided in the message header. Message data follows (in multiples of eight bytes). An eight-byte trailer contains an error indication code, which is zero if no transmission errors occurred between the source and destination nodes.

If the message data has been encrypted at the application level, the information in the MessageWay



(L2RH = L2 Routing Header, V = version, P = priority, E = endianness, PL = pad length, F = options flag)

Figure 4. MessageWay packet format (March 1996 draft [Cohen96]).

header is vulnerable to tampering. Changing the data length or endianness or destination address would effectively cause denial of service, since the resulting message would not be received correctly. The source address could be changed, so that the receiver is misled about the origin of the message. Perhaps most damaging would be to change the packet type: consider the effects of changing a read operation into a write.

To prevent tampering, the MessageWay header information may be authenticated. As described earlier, this involves combining a strong cryptographic hash of the header with some private information that proves the identity of the sender.

In some environments, authentication of the header is not enough. Even if an attacker cannot modify the packet, the address and type information in the header may be subject to traffic analysis. To defeat this sort of analysis, the original header information must be encrypted. This can be accomplished by encrypting the entire message and encapsulating the encrypted message inside another MessageWay packet. An observer on the outside will then only see "public" addresses and packet types. (As noted before, however, this may not protect against analysis of traffic patterns.)

Possible Security Extensions

The specific extensions provided by Secure MessageWay will not be described in this document. These will be described in a separate document, currently in draft form [George96]. Instead, this section will described the information that must be communicated among Secure MessageWay participants, along with mechanisms that may be used for that communication.

Security Context

Though MessageWay is a sessionless protocol, it may be convenient to establish a context in which participants exchange messages in a secure manner. This security context, or security association, is a list of attributes which are fixed for a given set of participants and which specify the specific security features used for exchanging a set of messages. Examples of the attributes which may be present in a security context are:

- *Cryptographic algorithms, protocols, and modes*. The parties must agree on the specific algorithms that they will use for cryptographic services, such as encryption and authentication. An example would be to use DES in cipher-block chaining (CBC) mode for encryption and RSA digital signatures for authentication. Also, any parameters that are constant over all its messages may be specified in the security context.
- *Lifetime*. A security context might expire after a certain amount of time, or after a certain number of messages have been exchanged. This limits vulnerability in the event that a given context is compromised.
- *Roles*. A context may be limited in the types of operations that it may support. For example, a context established for secure message passing may not be suitable for performing key distribution.
- *Keys*. A set of keys for cryptographic functions may be defined by the security context. These keys will be used to process all the messages associated with a context.

The alternative to establishing a security context is to communicate all the necessary information in each transmitted packet, protected as needed by a shared secret key or a public key mechanism. While both modes should be supported in Secure MessageWay, SHARE will use the security context approach, in order to avoid the cost of extracting information from each packet.

At the very least, a Secure MessageWay packet must contain a field identifying the security context to which it belongs. The security ID may be different for each participant in the context, since it may be difficult to negotiate a common ID for all. The packet, therefore, should contain the security ID associated with the receiver, since that is where the packet must be interpreted.

In addition to the security ID, there may be parameters that need to be carried in each packet. For example, a sequence number may be provided to prevent message replay or to be used as the initial value for a cryptographic operation. The presence or absence of these parameters will be dictated during the context negotiation (or will be implicitly agreed upon by all participants).

Options, Symbols, and Packet Types

There are currently two primary mechanisms under consideration for the transmission of security information in a MessageWay packet: options and symbols.

An option (or optional header) appears after the standard EEP header, as shown in Figure 4. A symbol, on the other hand, comes before the header, intermingled with the L2 Routing Headers. Using symbols is currently preferred, for the following reasons:

- *The symbol will always come before the data that it protects*. For example, consider the case in which the EEP header needs to be authenticated. With options, the header fields have already been read before the authentication key can be determined, so the header must be processed again for authentication purposes. If a symbol were used, the authentication of the header could be performed as it is being read the first time.
- *Symbols can easily indicate the scope of security functions*. For example, if secure sources are allowed to provide L2 source routes to secure destinations, the placement of the security symbol shows which L2RHs should be passed in the clear and which should be protected.[7] With options, L2RHs might not be supported; else they could be all received and then processed after the option specifies which should be protected, which is not compatible with wormhole routing.

Another alternative for conveying security information from one node to another is to always encapsulate the protected packet inside a new packet with a specialized packet type. The type extension field can be used to pass up to 16 bits of security information. If more security data is needed, it can be placed before or after the encapsulated packet in the security packet payload.

Message Authentication Code

Most security information should be located at the beginning of a packet, because it is used to initialize the cryptographic functions that act on the rest of the packet as it is received. The message authentication code (MAC), is an exception;. It is desirable to place the MAC at the end of the message, so that it may be computed as the rest of the message is being transmitted.

The MessageWay protocol is designed to support wormhole routing, where the head of a packet can be transmitted as soon as possible, even before the tail has been received. Consider a router, however, which accepts a packet from a secure SAN and prepares it for transmission over a non-secure SAN. If the MAC is placed at the beginning of the outgoing packet, then the entire packet must be received before it is transmitted. This prohibits wormhole routing and imposes a large latency and storage burden on the router, since a MessageWay packet can be hundreds of megabytes long. Therefore, it is imperative to transmit the MAC at the end of the secure packet.

[7] For the moment, we will ignore the covert channel available to a source node which can cause arbitrary routing information to be passed in the clear. The same arguments applies to other symbols that might be intended for the secure destination.

Options and symbols are always located at the beginning of a packet, so they are not suitable for carrying the MAC. There are two alternatives for placing the MAC at the end:

1. Encapsulated packet. If the protected packet is encapsulated inside a secure packet, the MAC can be placed at the end of the data block.
2. Optional trailer field. Currently, the only field allowed in the EEP trailer is the 64-bit error indication. If trailer options were supported, in addition to header options, these could be used for the MAC. Optional trailers are not included in the most recent draft specification [Cohen96].

To the MessageWay router which receives the secure packet, it doesn't matter whether the MAC is at the beginning or the end. In either case, it will compute the MAC as data arrives and compare its result to the transmitted MAC. There is a problem, however, with supporting wormhole routing on the receiving end: suppose a long packet arrives which fails authentication. The altered data may have already passed through the router and into the secure SAN. The destination node may in fact have already consumed the message before the problem was detected. (Note that this is a problem in regular MessageWay as well, since the error indication for a packet is carried in the trailer.)

There are two approaches to the problem. First, the receiving router can store the message until can verify its authenticity. This results in the same latency and storage problems described for the sender above. The practical result of this would be a restriction on the size of packets allowed for Secure MessageWay. On the other hand, the receiving node itself could be made responsible for not acting on a packet in a non-recoverable manner until it is determined that the packet was delivered reliably. A solution to this problem has not yet been chosen for SHARE.

SECURE MESSAGEWAY ROUTER

Figure 5 shows a generalized view of the SHARE network, consisting of four components: nodes, secure SANs, MessageWay routers, and the inter-SAN MessageWay network.

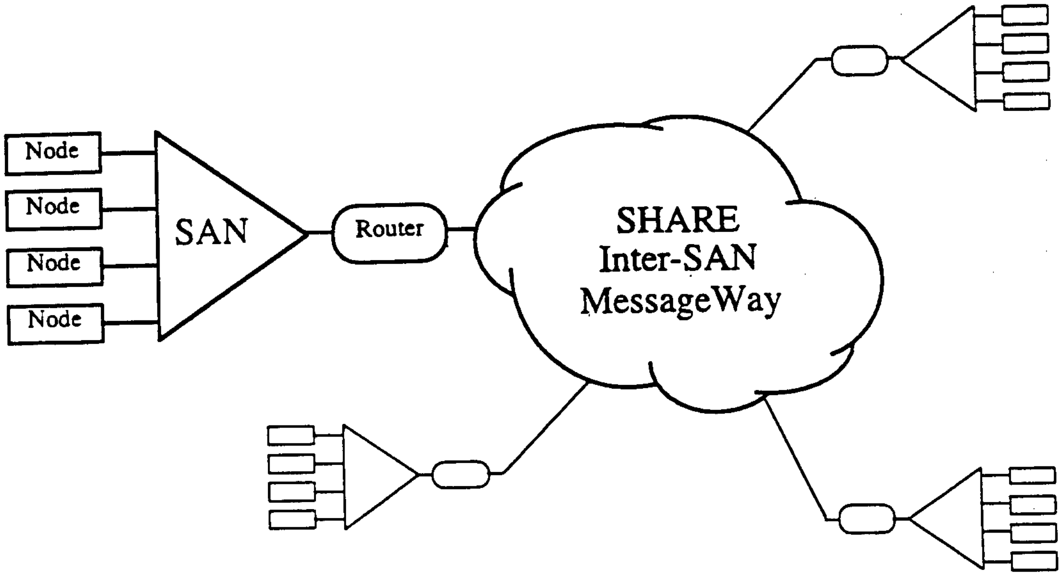


Figure 5. Generalized SHARE internetwork.

A node is a physical computing entity, which has one or more MessageWay addresses associated with it. (A MessageWay address may be used to denote either a logical entity, such as a process, or a physical entity, such as a device or a processor.) A secure SAN is a system area network, which is physically secure; all nodes in the SAN operate at the same security level. The Inter-SAN is non-secure and is subject to attack. It is composed of one or more SANs.

A MessageWay router is an entity that connects two SANs via the MessageWay protocol. It consists logically of two half-routers (HRs), each of which is responsible for routing packets on one of the SANs. (There may or may not be two physical HRs in the router.) For SHARE, one HR in the router is associated with the secure SAN and the other is associated with the Inter-SAN.

As described above, the SHARE approach to security is to provide a hardware cryptographic unit for high-speed cryptography. Secure MessageWay messages are used to deliver encrypted and authenticated messages between nodes, and to convey security information (such as security context IDs) between nodes and between crypto units as needed. There are three major alternate approaches for integrating the crypto units with the SHARE network. The crypto may be:

- integrated with each node,
- integrated with the MessageWay router, or
- a "bump-in-the-wire" on the inter-SAN side of the router.

Each of these alternatives was evaluated according to three criteria: performance, security, and portability. Based on that evaluation, the secure MessageWay router has been chosen as the favored approach for SHARE. The remainder of this section describes the secure router approach, including a high-level sketch of a Myrinet-based implementation. The other approaches are described and evaluated in Appendix B.

Router-Based Crypto

Heterogeneous, high-performance embedded computing demands support for various types of nodes and SANs, interconnected in a transparent, highly efficient manner. SHARE utilizes the MessageWay inter-SAN protocol for this purpose. The component which connects SANs together in a MessageWay is the router.

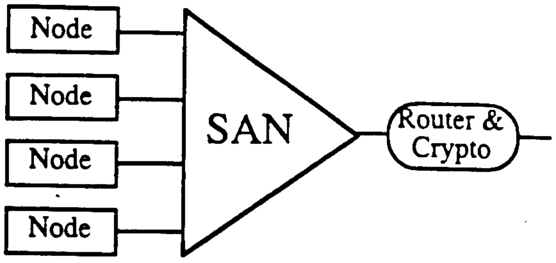


Figure 6. Router-based crypto.

The MessageWay router establishes a natural boundary between SANs. Any message that flows outside a local SAN must go through a router, which serves as both a physical and logical connection between SANs. Because of this well-defined boundary, it is logical to consider integrating the SHARE crypto unit with a MessageWay router, as shown in Figure 6.

Implementation

A MessageWay router consists of two half routers (HRs), each of which is responsible for one of the connected SANs. Each HR has its own MessageWay address; it may be a physical component, or it may be a logical process running on a processor. The two HRs that compose a single router are known as "twin HRs." Twin HRs are connected through some unspecified link. Examples of HR links include a standard I/O bus (e.g., PCI), a custom point-to-point link, or shared memory.

To implement the secure MessageWay router for SHARE, we assume that the HRs are separate physical entities. The cryptographic unit is placed on the link between the HRs, so that data is encrypted and authenticated as it flows from the HR on the secure SAN (the "red" HR) to the HR on the non-secure inter-SAN (the "black" HR). Similarly, data is decrypted and verified as it flows from the black HR to the red HR.

Figure 7 shows an implementation in which each SAN is Myrinet, and the inter-HR connection is PCI. Separate PCI busses will be used on each side of the crypto unit, in order to separate the red and black sides of the router. A separate connection to the crypto (not shown) is used to load keys and other security information.

The crypto unit maintains a table of state information, one entry per active security context. The state table contains the keys and other information needed to perform the cryptographic functions. An index into the crypto state table is known as a "tag." The tags associated with a given security context need not be the same at the sending and receiving cryptos; in fact, the tag used for encryption need not be the same as the tag used for decryption at the same crypto. The assignment of tags is performed as part of the initialization procedure for a security context.

To send a message from one secure SAN to another:

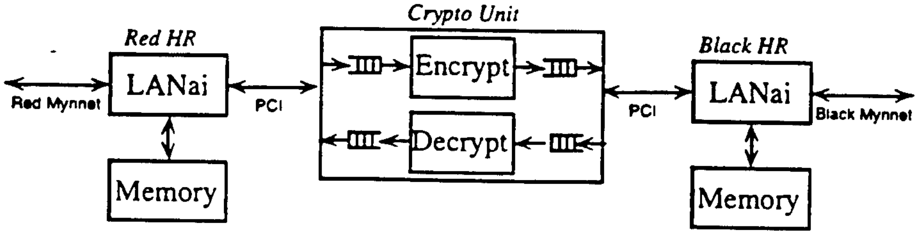


Figure 7. Myrinet- and PCI-based secure router.

1. The source node composes a MessageWay message, including an indication of the security context to which the message belongs. The message is delivered (via native SAN routing) to the red HR.
2. The red HR converts the security context ID to a tag, the index into the crypto state table. It may also determine whether the source and destination addresses are consistent with the security context.
3. The red HR passes the tag and the packet to the crypto unit. It may also prepend a known pattern to the packet to enable a quick initial verification of the decryption on the receiving end.
4. The crypto unit uses the tag to look up state information, including keys. The packet is encrypted, and a MAC is computed for the encrypted data. The encrypted data and MAC are passed on to the black HR, along with the tag and any other required information, such as a message sequence number.
5. The black HR uses the tag to look up routing information, which will include the MessageWay address of the black HR on the receiving end, and the tag for the receiving crypto. The black HR encapsulates the encrypted data and MAC into a secure MessageWay packet and transmits it to the destination black HR.

When the secure message is received at the destination router:

1. The black HR retrieves the tag and other security information from the packet. The encrypted data is passed to the crypto unit, along with the tag and other needed information.
2. The crypto unit uses the tag to lookup the decryption and authentication keys. A new MAC is computed on the encrypted data and is compared with the transmitted MAC to verify that the packet was not changed in transit. The data is decrypted and passed to the red HR, along with an indication of whether authentication passed or failed.
3. The red HR receives the original packet and routes it to the proper destination. It may also perform an initial verification check of the decryption by looking for the known pattern at the beginning of the decrypted data. If authentication fails, then the packet is dropped, or a bit is set in the packet's EEP trailer to indicate an error. (See the discussion on page 14.)

Performance

The performance of the secure router is highly dependent on the implementation of the HRs and on the connection between the HRs and the crypto unit.

There is very little actual routing that goes on the HRs. For an outgoing message, the red HR maps the security ID to a tag, and the black HR maps the tag to a route to the destination black HR. For an incoming message, the black HR retrieves the tag and the red HR maps the destination address to a route. These operations can easily be accomplished with a static table lookup, so the latency through the HRs should be minimal. Once the routing decision has been made, data should flow directly between the network and the crypto interface.

The major performance consideration is whether the data can be efficiently transferred between the SAN interface and the crypto interface. For highest performance, the bandwidth of the crypto interface should meet or exceed the bandwidth of the fastest SAN, and the latency associated with moving data from the SAN to the interface should be minimal.

If a suitable crypto interface can be found, then the added latency due to the cryptographic unit should be small.[8] The red HR can easily format the data according to the blocking requirements of the cryptographic algorithms, so the crypto unit merely needs to look up the crypto state for this message, initialize the cryptographic hardware, and pass the data through to the black HR.

The PCI local bus should meet the bandwidth requirements for SHARE. The 64-bit version of the bus, clocked at 66 MHz, results in a peak theoretical bandwidth of 528 MB/s; a full-rate Myrinet link has a peak theoretical bandwidth of 320 MB/s. The use of PCI would also provide portability, as discussed below.

It is worth noting here that the delay through the router would be incurred even without the cryptographic hardware, due to the nature of MessageWay. Even if only fixed L2-routes were used for all communication, each packet goes through MessageWay routers, which convert the first L2RH to the native SAN routing header for the subsequent SAN. The only performance penalty due to security with this approach is the delay through the cryptographic module and some indeterminate (but small) added complexity in the HR software.

Security

The security perimeter enforced by the router-based crypto exactly corresponds to the network architecture advocated by SHARE. All messages that flow from a secure SAN to a non-secure SAN must flow through the security unit. No MessageWay addresses or routing information relevant to the secure SAN will be exposed in the non-secure environment, because the entire MessageWay packet is encrypted. Finally, the security architecture is decoupled from the details of the SAN and inter-SAN network implementations.

All data flowing between the red and black SANs must flow through the crypto unit. The tag is the only data that flows across the unit without being encrypted. This represents a potential covert channel, but the use of this channel is mitigated in the following ways:

- The bandwidth is limited to one tag per packet; a tag is only a few bits, probably 16 or less. The very smallest MessageWay packet is 24 bytes, which translates into 48 bytes when encapsulated by the black HR. The tag represents at most about 4% of the network bandwith, or about 6.7 MB/s each way on a full-rate Myrinet link.
- The tag may not even be known by the source on the red side. The source knows the security context ID, and the red HR maps that to a tag, so the source can't manipulate the tag field directly. It can alter the field by using some number of context IDs. This reduces the effective bandwidth to only a few bits per message.
- Even if the mapping from context ID to tag is known for the red side, the actual value that gets placed in the black packet is the tag for the destination crypto, not the tag chosen by the red HR.

Other information, such as a message sequence number, may be included in the clear in the black encapsulated packet, but this information is supplied by the crypto unit itself, not a user process. The crypto, of course, must be completely trustworthy, since it contains keys. There must be no mechanism by which key information may be leaked to nodes on the red or black networks.

If the secure SAN is a single node, such as a workstation or PC, then the red HR may simply be a process that runs on the node, and the crypto is connected via PCI as described above. The black HR, not the node, directly controls the network and all interactions between the node and the black

[8] Estimates based on an existing ATM encryption system, using the same DES encryption chips described earlier, indicate a 1-2 microsecond latency. The portion of this estimate due to encryption and authentication is approximately 0.5 microseconds.

HR go through the crypto unit. Because of this separation between the red node (HR) and the black HR, we believe that the security of the router hardware may be certified independently of the node hardware and software.

The software implementing the red HR must be trusted, because it has access to the private contents of messages and because it selects the key which is used to encrypt each message. The keys themselves are not accessible to the red HR, but care must be taken to prevent the leaking of private data.

Key Management and Distribution

Keys and other state information are loaded into the crypto module by a trusted interface, not shown in Figure 7. This trusted interface is isolated from the red and black HR interfaces, so that neither has access to keys.

The simplest key distribution scenario is one in which the same set of keys is loaded into each crypto at boot time. Keys may be distributed through some convenient secure media, such as a Fortezza PCMCIA card.

Dynamic key generation and distribution is also possible. Suppose a key server, connected to the crypto module key interface, is also connected to the private SAN. Key servers may communicate among themselves in a secure manner by sending messages through the secure router (using a prearranged and statically loaded "master" key). The key negotiation and distribution messages would be encrypted and authenticated by the key servers themselves, so no key information would be visible to other members of the private SAN, and an observer on the public network would not be able to distinguish key message from any other kind of traffic flowing between the SANs.

In either key distribution scenario, a mapping must be devised between Security Context IDs, used by Secure MessageWay, and key indices used by the crypto module. This association between IDs and keys may be static or dynamic. The mapping is performed by the red HR.

Portability

Portability of the secure router depends on the choice of a standard interface between the HRs and the crypto and on the availability of interfaces that connect that standard interface to a variety of SAN types. PCI would appear to be a good choice, given its potential performance and its availability on a range of platforms.

Summary

The MessageWay router offers a natural integration point for the cryptographic functions required by SHARE and Secure MessageWay. In order for high-performance MessageWay to exist at all, there is a need for routers which offer high-bandwidth connectivity between SANs. These routers will in turn rely on high-performance network interfaces based on a small set of common standards for intra-router (inter-HR) communication. By basing the SHARE cryptographic module on one of these standards, we can impose a clean architectural separation between secure and non-secure SANs without sacrificing generality or performance.

APPENDIX A: BASIC CRYPTOGRAPHY

In this appendix, we provide a brief introduction to cryptographic operations and protocols, used to implement the security features described in this report. *Applied Cryptography* [Schneier96] provides more detailed information, as do many introductory books on cryptography.

Privacy

One of the major requirements for secure communications is privacy, the property that a message may not be read by any unauthorized party. (An unauthorized party is also known as an "attacker" or "intruder.") More formally, an unauthorized party should be able to gain no information about a private message except for its length.

Data privacy is accomplished through the use of encryption, in which a mathematical function (the cryptographic algorithm or cipher) is applied to the original text (plaintext) to produce encoded text (ciphertext). The algorithm depends on a value called the key; only parties that know the value of the key can decrypt the message (convert it from ciphertext back to the original plaintext). It is generally assumed that the attacker knows the algorithms or protocols used for encryption; only the key value is unknown.

The security of an encryption algorithm depends on how difficult it is to determine the key. If the range of key values is small, an attacker can just decrypt the message with every possible key value until the resulting plaintext makes sense. (This is known as a "brute force" attack.) Other types of attack generally involve examining many samples of ciphertext, or known ciphertext-plaintext pairs, to recover the plaintext or key. An algorithm is considered secure if the cost of the best known attack is considered impractical or infeasible for the presumed attacker. (Note that what may be impractical for your next-door neighbor may be routine for a government agency.)

Symmetric key encryption

In one form of encryption, the same key is used to both encrypt and decrypt the message.[9] The key is shared between the sender and receiver of the message and must be kept secret. This is known as symmetric key (or secret key) encryption. These ciphers involve permutations and substitutions of the plaintext to generate the ciphertext. An example of a symmetric key algorithm is DES [Schneier96, ch. 12].

Public key encryption

Another type of encryption involves the use of two distinct but related keys, one used for encryption and the other for decryption. One of the keys is made public, but the other, known as the private key, is kept secret by its owner. If Alice wants to create a message that only Bob can read, she encrypts it with Bob's public key. Only Bob can decrypt the message, since he is the only holder of the private key. The security of a public key system rests on the difficulty of calculating the private key, given the public key. These algorithms generally involve mathematical operations based on very large integers with very large prime factors; these algorithms are typically much slower than symmetric key algorithms. An example of a public key cryptographic system is RSA [Schneier96, sect. 19.3].

Authenticity

The second major requirement for secure communications is authenticity, the property of knowing that a message received is exactly the same message that was sent and that the claimed sender is in fact the actual sender. This involves generating some sort of "fingerprint" of the message data,

[9] More generally, the decryption key can be easily calculated from the encryption key, and vice versa.

which could only have been computed by the sender, and which can be easily verified by the receiver. (A related concept is non-repudiation, in which the sender cannot later deny having sent the message.)

Digital signature

Public key algorithms can be used to sign a message, that is, to associate a value based on the data that only the signer could have generated.[10] For example, suppose Bob encrypts a message with his private key and sends both the original and encrypted messages to Alice. Only Bob could have generated the encrypted message, and Alice can easily decrypt it using Bob's public key to verify the signature. In practice, since public key encryption of large documents consumes a lot of time and space, the signer first generates a message digest, using a cryptographic hash function, and then encrypts the digest rather than the entire message. The RSA public key algorithm can be used for digital signatures; DSA [Schneier96, sect. 20.1] is a public key algorithm designed for signatures only, not for encryption.

A cryptographic hash function (also known as a one-way hash) condenses an arbitrary-length text into a fixed-size digest with the following properties: (1) It is computationally infeasible to find any message that corresponds to a particular digest. (2) It is computationally infeasible to find two message that hash to the same digest. An example of a strong cryptographic hash function is MD5 [Rivest92].

Message authentication code

Digital signatures provide authentication which can be verified using public information. If two parties trust each other enough to share a common secret, then a message authentication code (MAC) can be generated using the secret value. MACs are based on cryptographic hash or symmetric key encryption operations, which are faster than the public key operations needed for digital signatures. Examples of MACs are keyed MD5 [Metzger95] and the family of XOR MACs [BGR95].

[10] Symmetric key algorithms may also be used for digital signatures, but this requires the participation of a trusted third party.

APPENDIX B: ALTERNATIVE ARCHITECTURES

In this appendix, we discuss two alternative security architectures that were considered for SHARE: a node-based crypto module and a "bump-in-the-wire" crypto module. These approaches were abandoned during initial tradeoff studies but could be useful under different system requirements.

Node-based Crypto Unit

The first alternate approach, shown in Figure 8, is to associate a cryptographic module with each node on the secure SAN. A node, in this context, is a physical computing entity which is the "home" for one or more MessageWay addresses. Each node is responsible for formatting its own Secure MessageWay message when communicating with nodes in another secure SAN. The crypto module provides encryption and authentication services for use when sending and receiving messages.

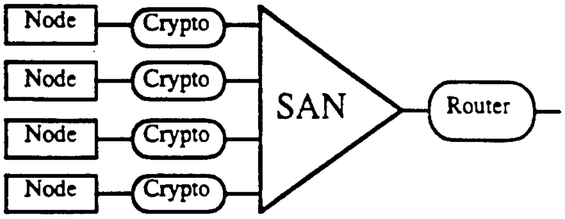


Figure 8. Node-based crypto.

In this scenario, the sending node would create the proper MessageWay header to indicate the keys used to encrypt and authenticate the data block, as well as any other information needed by the receiving node to verify and decrypt the message. The MAC would need to be included as a part of the transmitted packet, preferably at the end of the packet, that it may be computed as data is moved from the node to the network.

Implementation

One possible implementation of the node-based approach is to design the crypto module as a peripheral device, as shown in Figure 9. The crypto hardware is available directly to the processor, and possibly to the network interface. The crypto services are thus made available to any software layer, including applications. The figure shows the crypto unit attached the processor's I/O bus, like the network interface. The crypto could also be connected to the memory bus, or the "bus" could be a SAN that connects processors, memories, and I/O devices.

An alternative, shown in Figure 10, is to integrate the crypto with the network adapter. In this case, data passes through the crypto unit as it is transmitted or received by the network adapter. In the case of the figure, the sending node writes data to the address associated with the encryption unit, rather than directly to memory. The encrypted packet is read from the LANai local memory and sent to the network. When data is received from the network, the LANai writes it to the decryption device, and the node reads the decrypted packet from memory.

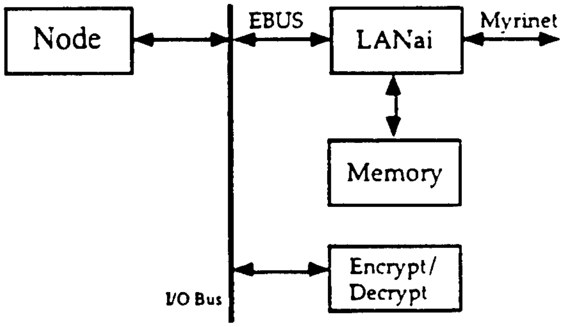


Figure 9. Crypto on I/O bus.

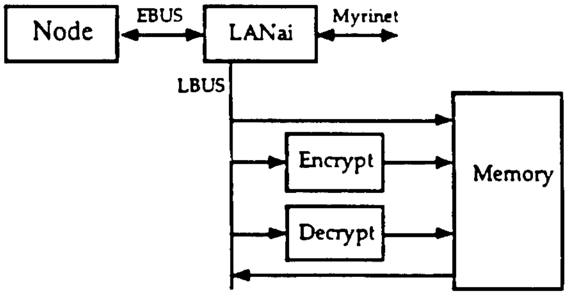


Figure 10. Encrypting network adapter.

Performance

The peripheral-based implementation (Figure 9) has a potential performance penalty, because data must traverse the I/O bus at least twice. First, the node would move data from the memory into the crypto unit, where it would be (for example) encrypted and authenticated. The resulting data would then be moved either back to memory or directly to the network interface. As a result, the bandwidth requirements for the I/O bus are increased, and the latency through the crypto module involves multiple I/O transfers.

The performance of the adapter-based approach (Figure 10) depends on the degree to which the crypto unit is integrated with the network transfer hardware. In the LANai example shown, the only additional latency is introduced by the cryptographic hardware itself, so this configuration should result in low latency and high bandwidth. On the other hand, if the crypto unit was simply placed in front of an existing network adapter, using a standard interconnect like PCI, then some performance would likely be lost in communicating through the general I/O interface.

Security

The consequence of implementing security at each node is that there is no clear delineation between the secure non-secure SANs. Instead, there are secure and non-secure nodes, and every SAN is treated as a non-secure SAN. This has the following undesirable implications:

- *Every* message between nodes must be a secure message. If the security perimeter is at the node, then the network adapter cannot allow data to pass through it unprotected. The result is that there can be no distinction between intra-SAN and inter-SAN messages, even if the local SAN is considered physically secure. This imposes a latency penalty on every message between nodes.
- L2 routing must be prohibited. If secure messages pass directly between nodes, then all routing information must be in the clear, so that routers may act on it. If L2 routing headers are used, then an attacker can learn about the topology of secure SANs by intercepting messages and interpreting the L2 headers.
- Traffic analysis is made easier by the fact that source and destination addresses which appear in the secure packet must be associated with the actual source and destination nodes. This can be mitigated to some extent by using logical addresses to mask the true addresses of the sender and receiver or by not providing the source address.

In addition, the close integration of the crypto hardware with the host could make it more difficult to pass security certification. In the LANai-based system, for example, it would be possible for the node to bypass the cryptographic hardware or to cause a decrypted message to be sent back out on the network in its plaintext form. The networking software on the node would have to be carefully evaluated to make sure that it couldn't fail or be circumvented in ways that would allow protected data to leak out. This would make certification more difficult, if not impossible, and the system would have to be recertified with every change in node software.

Portability

The node-based crypto is very dependent on details of the node I/O system and on the SAN to which it is connected. This can be mitigated by using a standard, widely-used I/O architecture and SAN, but that still places limits on the environments in which a single design can be directly applied.

Summary

It appears that a node-based crypto unit could be designed which would support high-bandwidth network connectivity with little additional latency. However, the security model supported by the this architecture does not match well with the inter-SAN security required by SHARE. In addition, the crypto unit design is node-specific and SAN-specific, which is not in keeping with the heterogeneous nature of SHARE.

Bump-in-the-Wire Crypto

A second alternative involves completely separating the crypto module from the router. Instead, the crypto module establishes a security perimeter on the public side of the MessageWay router, as shown in Figure 11. The crypto module translates packets between secure and non-secure parts of the inter-SAN network. The MessageWay routers are responsible for routing packets through crypto units to implement the security functions. Since the crypto units don't initiate, receive, or route messages, they appear as "bumps" in the inter-SAN network (but hopefully not "speed bumps").

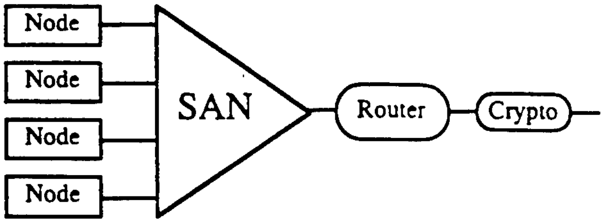


Figure 11. Bump-in-the-wire crypto.

One advantage of separating the crypto and routing functions is that a crypto unit may be shared among several routers. Consider, for example, several SANs connected via Myrinet, all in a secure environment. A bump-in-the-wire crypto unit is located at the boundary of the secure environment; all messages that leave the environment must go through the crypto. Each router knows how to route outgoing messages to the crypto and how to convey security information so that the proper operations happen at the crypto. The number of cryptos, however, is independent of the number of routers. The specific configuration is determined by performance, connectivity, and economic factors.

Implementation

In some ways the implementation of the bump-in-the-wire (BIW) crypto is similar to the router-based crypto: data flows through in a pipelined fashion, getting encrypted or decrypted and authenticated along the way. The standard inter-HR interfaces of the router-based crypto, however, are replaced with network interfaces. Another difference is that some routing information in the packet needs to flow through the crypto unchanged, so that network on the public side can route the packet to the receiving crypto.

Figure 12 shows a possible Myrinet-based implementation of the BIW crypto. Because it is known that the network interfaces on both sides of the bump will be the same, a tight integration of the two is used to minimize latency and maximize throughput. In the example shown, the two LANai's are coupled by the simple, high-bandwidth LBUS. As a packet arrives at the LANai from

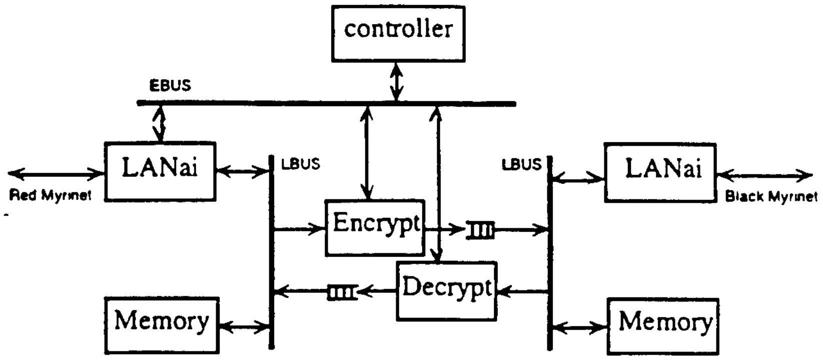


Figure 12. Myrinet-based BIW crypto.

the network, it is written directly to the crypto hardware via the LBUS, rather than to memory. Similarly, an outgoing packet is moved directly from the crypto hardware to the packet interface.

The control program running on the LANai processor is very simple. It only needs to know how to interpret the first part of an incoming packet to retrieve security information. It then initializes the crypto hardware and transfers the packet data at a high rate from the packet interface to the crypto hardware. It may also be required to detect key management packets and relay those to the on-board controller via the EBUS interface. The controller is responsible for initializing and maintaining the information in the crypto state tables.

The final implementation issue is whether the crypto unit interprets packets at the data link layer (Myrinet, in this case) or at the network layer (MessageWay). In the first case, security information is embedded in the Myrinet frame, and the MessageWay packet that flows between routers is not affected at all. In the second case, the incoming Myrinet packet is interpreted as a MessageWay packet, and all security transformations happen at the MessageWay level (i.e., the outgoing packet is just a regular Myrinet packet).

The transformation of a packet in a Myrinet-based system is shown in Figure 13. A reserved Myrinet packet type is used to signal the beginning of the security info, which is embedded in the regular Myrinet routing header. If a crypto receives a packet which does not start with security info, then the packet has been misrouted and should be dropped.

After the security info is the Myrinet routing used to transmit the packet across the inter-SAN Myrinet to the receiving crypto. This routing header is stripped off, one byte at a time, as it passes through Myrinet switches. When it arrives at the destination crypto, the first item in the packet will be the security info necessary to decrypt and verify the packet. After the second security field, there may be additional routing bytes (not shown in the figure) to deliver the packet to the destination router. The Myrinet packet data contains the complete MessageWay packet, as transmitted by the source router. The data is followed by the one-byte Myrinet checksum (CRC). The end of the packet is designated by the Myrinet GAP symbol (not shown).

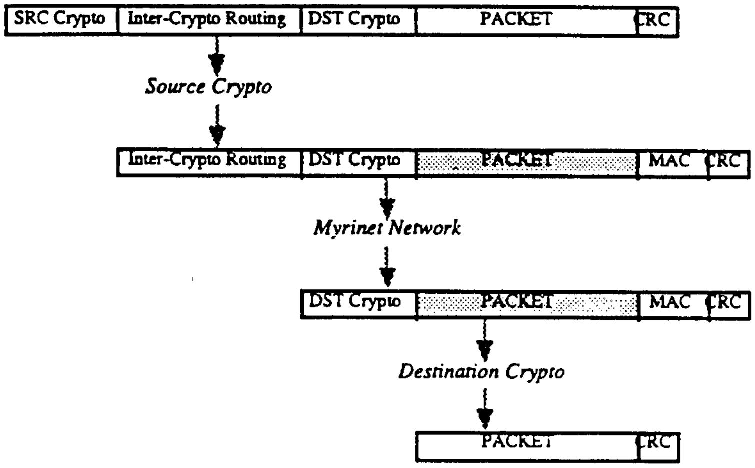


Figure 13. Packet flow for Myrinet BIW.

When a packet is received from the red side, the source crypto does the following:

1. The security field is removed and used to lookup the crypto state required for this packet.
2. The routing bytes and destination security field are passed through the crypto hardware unchanged.
3. Everything following the second security field is encrypted and authenticated. The encrypted data is transmitted on the black side.

4. The MAC is appended to the end of the packet, before the Myrinet checksum. The checksum must be computed over the entire encrypted packet, including the headers and the MAC.

The packet is then routed as a normal Myrinet packet. When it arrives at the destination crypto:

1. The security field is removed and used to lookup the crypto state required for this packet.
2. The remainder of the packet, up to the MAC, is piped through the authentication and decryption hardware. The transmitted MAC is compared against the computed MAC. (The size of the MAC must be known by the destination crypto, so that it may be distinguished from the encrypted data.)
3. The MAC is stripped off as the packet is transmitted to the destination router.

One problem with the data link layer approach is that it requires a homogeneous inter-SAN network. In the case described above, the inter-SAN network must be a single Myrinet network, because all inter-crypto routing and security information must be embedded in the Myrinet header.

For this reason, we now consider a MessageWay-based approach. Figure 14 shows a possible packet transformation if the BIW crypto operates at the MessageWay layer. The operations are similar to the Myrinet case described above, but the implementation is slightly different.

The first item in the MessageWay packet is a symbol, which carries the security information needed by the source crypto. This symbol is stripped from the packet and used to lookup crypto state for this packet.

Following the security symbol is the inter-crypto routing information, in the form of one or more L2 routing headers. Note that this allows for the possibility of a mixed inter-SAN network, with intermediate MessageWay routers, rather than the one-level Myrinet approach implied above.

At the end of the inter-crypto routing headers is another symbol, which carries the security info for the destination crypto. Additional L2RHs may be provided for routing the final packet to the destination router (or all the way to the destination node). At the source crypto, everything after the second security symbol is encrypted and authenticated. The result is encapsulated in a new MessageWay message, similar to the router-based crypto described above. Encapsulation is necessary for two reasons:

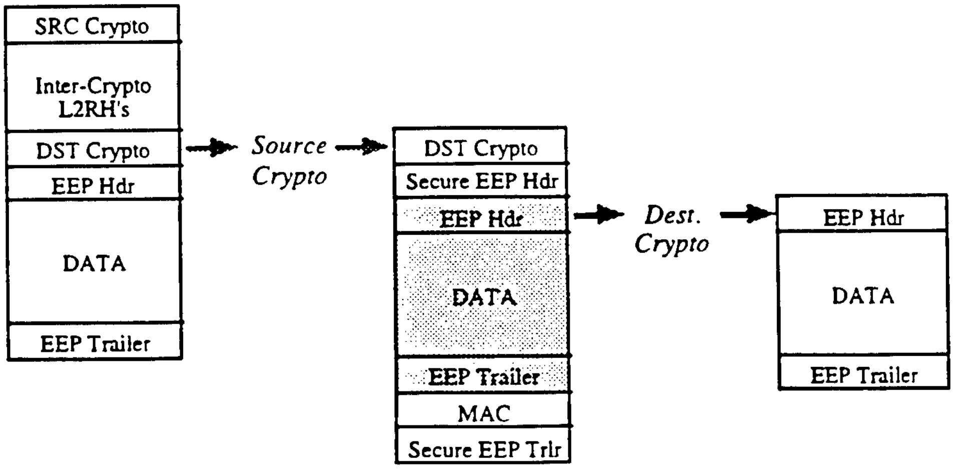


Figure 14. Packet flow for MessageWay BIW.

1. The MAC must be provided at the end of the packet, and there are no currently proposed mechanisms for adding data at the end of a MessageWay packet.
2. The packet may pass through intermediate MessageWay routers; the length of the encrypted packet is required at each router.

An additional benefit of encapsulation is that the original source, destination, and type of the packet are hidden, thwarting attempts at traffic analysis.

Performance

In both the data link and network layer approaches, the underlying data transfer hardware is the same. The BIW approach is attractive because the hardware can be optimized for the specific SAN type chosen. Therefore data can flow efficiently from one interface to the other, through the high-speed cryptographic hardware.

Latency, however, will be affected by the choice of data link or network layer. Manipulating data at the data link layer is somewhat easier, since it is closer to the form which is manipulated by the network interfaces themselves. The method for conveying security information can be adapted to suit each data link protocol (e.g., packet type for Myrinet, VPCI for ATM). Changes in the packet structure need not cause changes in the packet data (e.g., the MessageWay data length field).

A network layer implementation, however, requires that each packet be converted into a network layer packet, which could involve reassembly of data link packets or other sorts of interpretation. Transformations at the network layer would then have to be reflected back down to the data link layer (e.g., via segmentation) for the outgoing transmission.[11]

Security

Both the data link layer and network layer versions of the BIW crypto establish a firm security boundary at the entrance to the public inter-SAN network. Both encrypt the entire original MessageWay packet, including routing information for the secure SANs, and therefore provide protection against traffic analysis.

One consequence of separating routing from crypto is that the crypto must pass routing information in the clear from its red side to its black side. Because this routing information is of arbitrary length, this represents a significant potential covert channel. It also makes the system more vulnerable to failure; if a Myrinet-based BIW, for instance, somehow fails to detect the second security info field, sensitive data which follows that field would be inadvertently leaked into the black network. The only apparent way to avoid these problems is for the crypto to generate its own routing information based on the security tag, which in effect reinvents the router-based approach described above.

An advantage of the BIW approach, on the other hand, is that the router is completely within the security boundary. This means that the HRs can freely exchange routing information in support of dynamic routing and discovery. Because the HRs on the public side of the router communicate through cryptos, pre-assigned keys can be used to insure that routers can exchange information in a private and authenticated manner.

[11] Segmentation and reassembly are, of course, not issues for a Myrinet-based design, but the architectural decision to provide a network-layer BIW could have significant performance implications for other network implementations.

Portability

In order to sustain high bandwidth and low latency, the BIW crypto will likely be designed for a specific network implementation. If a Myrinet-based solution were designed for SHARE, it woulc not be directly applicable to an environment in which ATM or FDDI or SCI is used as the inter-SAN connection. Significant parts of the cryptographic hardware would be reusable, but not the interaction of the crypto hardware with the network interface.

On the other hand, the BIW architecture allows existing data link layer encryption devices to be used in place of a SHARE-designed Myrinet solution, if desired. For example, if the Myrinet inter-SAN is replaced with ATM, then the MCNC Enigma2 [Stevenson95] or GTE's FASTLANE [GTE] could be used with very little change to the MessageWay infrastructure. This is a means for introducing the SHARE software into an existing secure networking environment (but only if the data link layer approach is used).

Summary

The bump-in-the-wire crypto approach is attractive because of its potential for highly optimized performance, and because of its flexibility in terms of network configuration and dynamic routing. There is a major security concern, however, with allowing arbitrary amounts of routing information to pass through the crypto unaltered.

Providing security at the data link layer offers high performance, but the design becomes tied to a specific implementation of a specific network. Providing security at the network layer offers increased generality, especially with regards to the structure of the inter-SAN network, but forces the unit to be less like a bump and more like a gateway, with the associated increase in latency.

REFERENCES

[Atkinson95] R. Atkinson. *Security Architecture for the Internet Protocol.* Internet RFC 1825. August 1995.

[Aziz95] Ashar Aziz. *Simple Key-Management for Internet Protocols (SKIP).* Internet Draft (work in progress), IPSEC Working Group, IETF. November 1995.

[BGR95] Mihir Bellare, Roch Guerin and Phillip Rogaway. "XOR MACs: New Methods for Message Authentication Using Finite Pseudorandom Functions." *Crypto '95.* 1995.

[Boden95] Nanette J. Boden, et al. "Myrinet -- A Gigabit-per-Second Local-Area Network." *IEEE Micro.* February 1995.

[Cohen96] Danny Cohen and Craig Lund. *Proposed Specification for the MessageWay Protocol.* Internet Draft (work in progress), MsgWay Working Group, IETF. March 1996.

[Dai96] Wei Dai. Posting to sci.crypt, Usenet newsgroup. Jan 18, 1996.

[Dally87] William J. Dally and Charles L. Seitz. "Deadlock-Free Message Routing in Multiprocessor Interconnection Networks." *IEEE Transactions on Computers*, vol. C-36, no. 5, May 1987, pp. 547-553.

[Eberle93] H. Eberle. "A High-Speed DES Implementation for Network Applications." *Lecture Notes in Computer Science: Advances in Cryptology -- Crypto '92 Proceedings.* Springer-Verlag. 1993.

[FIPS113] National Bureau of Standards, US Department of Commerce. *Computer Data Authentication.* Federal Information Processing Standards Publication 113 (FIPS PUB 113). May 30, 1985.

[George96] Robert George and Tony Skjellum. Secure MessageWay design document, in progress.

[GTE] GTE Government Systems Corporation. FASTLANE ATM Encryptor (KG-75). Information on World-Wide Web page available via <http::/www.gte.com>.

[Karn95] P. Karn and W. A. Simpson. *The Photuris Session Key Management Protocol.* Internet Draft (work in progress), IPSEC Working Group, IETF. November 1995.

[Maughan95] Douglas Maughan and Mark Schertler. *Internet Security Association and Key Management Protocol (ISAKMP).* Internet Draft (work in progress), IPSEC Working Group, IETF. November 1995.

[Metzger95] P. Metzger and W. Simpson. *IP Authentication Using Keyed MD5.* IETF RFC 1828. August 1995.

[MPI94] Message Passing Interface Forum. "MPI: A Message-Passing Interface Standard." *International Journal of Supercomputing Applications*, vol. 8, no. 3/4, 1994.

[Myricom96] Myricom, Inc. Myrinet Link Specification. Unpublished. Available from <http://www.myri.com/myrinet/products/documentation/link/>.

[Rivest92] Ron Rivest. *The MD5 Message Digest Algorithm.* Internet RFC 1321. April 1992.

[Schneier96] Bruce Schneier. *Applied Cryptography, Second Edition.* John Wiley and Sons, 1996.

[Stevenson95] Daniel Stevenson, et al. "Design of a Key Agile Cryptographic System for OC-12c Rate ATM." *Internet Society Symposium on Network and Distributed System Security*. IEEE Computer Society Press. 1995.

[TIS96] Trusted Information Systems, Inc. *Worldwide Survey of Cryptographic Products*. <http://www.tis.com/crypto/survey.html>. January 1996.

[Touch95] Joe Touch. "Performance Analysis of MD5." *Crypto '95* . 1995.

[Wiener93] Michael J. Wiener. "Efficient DES Key Search." August 1993.

DRAFT

January 1997

Proposed Specification for Security Extensions to the PacketWay Protocol

Robert George, MSU
with considerable help from
Jeff Smith and Fred Shirley, Lockheed Sanders
Tony Skjellum and Thom McMahon, MSU
Greg Byrd, MCNC
and Danny Cohen, Myricom

PacketWay-WG

Please send your comments re this draft to <robert@erc.msstate.edu>.

Part-1: Secure PacketWay EEP messages

This memo describes Secure PacketWay -- an applique over the PacketWay protocol utilizing security concepts necessary for providing cryptographic key management and secure communication contexts in a PacketWay (here thereafter referred to as "PktWay") environment. Secure PktWay provides a protocol that negotiates, establishes, modifies and deletes Security Contexts in high-performance System Area Networks (SANs) and high performance Local Area Networks (LANs), where there will be numerous security mechanisms and several options for each security mechanism. It is important that the key management protocol be flexible enough to provide both public key generation and distribution, as well as private key management. Secure PktWay defines the procedures for authenticating a communicating peer, key management, and threat mitigation (e.g., denial of service and replay attacks). All of these are necessary to establish and maintain secure communications between trusted System Area Networks separated by untrusted Local Area Networks.

This part describes security extensions to the EEP (End-to-End Protocol) PktWay-protocol. A special type of EEP message constitutes the RRP (Router-to-Router Protocol) which is described in Part-2, and their format in Part-3.

Some basic Secure PktWay terminology requires explanation. Secure PktWay interconnects "trusted" (or secure) System Area Networks (SANs) by routing data across "untrusted" (or unsecure) networks. Each secure SAN is comprised of trusted nodes, which communicate within the SAN using local security policies. At least one node in each secure SAN is also a Secure PktWay router, connected through 0 or more untrusted SAN's to at least one other secure SAN.

For the purposes of this document, network security is defined as the set of services used to assure the secure delivery of data between secure SAN's. Secure delivery means that data is not delivered to unauthorized parties and that data may not be altered during transmission between the source and the intended destination without detection. Sec PktWay provides network security through the use of security protocols intended to facilitate cryptographic functions.

The assumed threat on the inter-SAN network is an active adversary which has physical control of one or more network links. Any data that traverses compromised links may be read by the adversary, and data may be arbitrarily inserted, removed, or replaced on the link. It is the purpose of Secure PktWay to insure that an attacker cannot interpret messages sent from one secure SAN to another, or create a message that will be accepted as genuine by a secure SAN, either by forgery or by replaying a previous legitimate message.

PktWay-Msgs <04> PktWay-WG

THE BASIC MODEL

The basic model of Secure PktWay is a set of SANs (System Area Networks), each with its own conventions and protocols.

The interconnection between at least two of the SANs will be via Secure PktWay-routers. A router between the SAN-A and the SAN-B is composed of two interconnected processes, each a fully-fledged node on a SAN. These processes are known as HRs ("Half-Routers") or "SAN-interfaces."

Secure PktWay is based on secure ("red") nets (SANs and/or LANs), interconnected via a non-secure ("black") nets, all using PktWay. The communication over the black nets is performed by encapsulating encrypted red PktWay messages inside unencrypted black PktWay messages. As always in PktWay, routers between nets are made of two HRs. However, routers between red and black nets have a red-HR, a black-HR, and an encryption/decryption device between them. Thus Secure PktWay packets travel through black SANs encapsulated inside plaintext PktWay packets, and are transported by the native packet format of each SAN, by being prefixed with the routing header and followed by the tail as required by that SAN. For all practical purposes the existance of the black-nets is transparent to the red nets, and no communication is possible between entities on red and black nets.

```
    Thus the following configuration:
                +-----+ +-----+                        +-----+ +-----+
    Red-SAN-A---+HRA1+E+HRB1+---Black-SAN-B---+HRB2+E+HRC1+---Red-SAN-C
                +-----+ +-----+                        +-----+ +-----+


    cannot be distinguished (by hosts on the red nets) from:

                +-----+                                +-----+
    Red-SAN-A---+HRA1+--------------------------------+HRC1+---Red-SAN-C
                +-----+                                +-----+
```

Since the L2 source routing information that preceeds a PktWay header is considered sensitive information (which describes the network architecture of the secure SAN), it should not be exposed to the untrusted black inter-connection network. The Secure PktWay router must therefore provide translation of secure L2 source routes to unsecure routes which contain only the hops necessary for packets to travel from one black HR to another black HR. Similarly, the Secure PktWay router must also provide translation from secure to unsecure L3 addresses.

NOTATIONS

8B means "8-byte"

0x indicates hexadecimal values (e.g., 0x0100 is 2^8=256-decimal).

0b indicates binary values (e.g., 0b0100 is 4-decimal).

xxx indicate a field that is discarded without any checking (e.g., padding).

The Secure PktWay MESSAGE STRUCTURE

Secure PktWay utilizes the same packet format as PktWay. Thus messages have 6 components, including 4 optional ones:

[1] [Optional Sequence of L2-Routing-Headers Records (L2RHs) and Symbols]	
[2] EEP Header (16 bytes)	(PH)
[3] [Optional Header fields]	(OH)
[4] [Optional Data Block]	(DB)
[5] [Optional Trailer fields]	(OT)
[6] EEP Trailer (8 bytes)	(TAIL)

In contrast to PktWay, it is expected that most Secure PktWay messages will have Optional Header fields as well as Optional Data Blocks.

[1] Optional Sequence of L2-Routing-Headers Records (L2RHs) and Symbols

Secure PktWay exploits the Optional Symbol which may preceed the PktWay EEP header. Secure PktWay uses the Optional Symbol field to contain a variable-length Security Context Identifier (SCID). This SCID indicates to the local Secure PktWay router that the packet contains data intended to be routed across an unsecure network to the SCID peer, which consists of a destination node within a foreign SAN.

The Security Context is requested by a secure node upon initiation of traffic between secure SAN's. The request is received by the local Secure PktWay router and negotiated with the Secure PktWay router representing the local nodes' secure peer. The negotiation between the routers includes the choice of authentication method, encryption scheme, and key type. The Security Context resulting from the negotiation is referenced by a Security Context Identifier (SCID), which is returned to the node requesting the Security Context.

Subsequent traffic from the secure source node to the secure destination node is pre-pended with the negotiated SCID symbol.

The L2 header must contain enough information to allow a router to quickly create any necessary local routing headers and trailers. Secure PktWay implementations that support L2-forwarding must document their unique L2 header requirements.

When a Secure PktWay message is encapsulated inside any native SAN message (Paragon and Myrinet, for example), it's up to that SAN to distinguish between it and other native packets. This is not a Secure

PktWay issue. For example, Myrinet uses its Message-Type to recognize Secure PktWay messages.

Secure PktWay-Routers on the boundaries between secure and unsecure SANs are asked to forward packets with either L2 or L3 routings. The former start with an L2RH, (having both its 9th and its 10th bits set to 1), whereas the latter start with Secure PktWay-addresses (with other values for these 2 bits).

FORMAT:

Each Secure PktWay Symbol is in the format:

```
+--------+--------+--------+--------+--------+--------+--------+--------+
|vv000000|1011ssss|ssssssss|ssssssss|LLLLLLLL|  SCID  |  ....  |  xxx   | Symbol
+--------+--------+--------+--------+--------+--------+--------+--------+
                <----- Symbol-ID ----> <-- L -> <----- SCID ---->
```

The first 2 bits are 0b00 for the working version of the protocol. They should have other values for experimental versions.

The next 6 bits should be all zeroes.

The next four bits must be 0b1011 to indicate that this is a Symbol record. The next 20 bits are the Symbol Identifier, followed by an 8-bit byte count of the Security Context Identifier. SCID information starts in the sixth byte, and is followed by as many padding bytes as needed to fill to the next 8B-boundary. This restricts the length of the SCID not to exceed 255 bytes. This length is expected to be greater than zero.

EXAMPLES:

A Symbol with a 2 byte SCID:

```
                  5                                  #1       #2    padding
+--------+--------+--------+--------+--------+--------+--------+--------+
|00000000|10110101|00100001|01010001|00000010|  SCID  |  SCID  |  xxx   | Symbol
+--------+--------+--------+--------+--------+--------+--------+--------+
                <----- Symbol ID ----> <-- L -> <---- SCID ----->
```

[2] EEP Header (16 bytes) (PH)

```
 2   6                24                       16                 16
+-+------+--------+--------+--------+--------+--------+--------+--------+
|V|  P   |     Destination-Address           | Type-extension | Packet-type     |PH1
+-+-+---++-----------------------------------+-+------+--------+-----------------+
| E | PL|    Data-Length (8B-words)          |h|  RZ  |      Source-Address      |PH2
+---+---+--------+--------+--------+--------+-+------+--------+--------+--------+
  4   3               25                     1   7             24
```

These fields are described below.

[3] Optional header fields (OH)

Secure PktWay uses Optional Header fields (OH) to contain data and parameters necessary for cryptographic functions, as well as data used to authenticate the EEP header itself. As an example, the optional header may contain long-cycle chaining information, or block cypher Initial Value data.

Secure PktWay's use of Optional Header fields requires that the Option-Flag (h) is set to 1 in the EEP-header.

Each OH is in the format:

```
+--------+--------+--------+--------+--------+--------+--------+--------+
|1Ctttttt|LLLLLLLL|  data  |........|........|........|........|........| OH
+--------+--------+--------+--------+--------+--------+--------+--------+
```

The first byte indicates the TYPE of this optional header field (OH-TYPE).

The first bit of the first byte must be 0b1, to indicate that the Secure PktWay Optional Header is mandatory, and must be processed.

The second bit of the first byte, C, indicates that there are no more trailer fields (i.e., this is the last field of this message).

```
 C=0: More trailer fields follow
 C=1: End of Optional fields group (OH or OT)
```

The next 6 bits indicate the TYPE of the Secure PktWay Optional Header field. Secure PktWay adds the following Optional Header types:

```
   0x06: MAC here
   0x07: MAC following the DB
   0x3E: Cryptographic Data
```

The second byte is the byte count of the data for this field that starts in the third byte, and is padded with as many padding bytes as needed to fill 8B-words.

Example:

```
                   4      #1       #2       #3       #4    padding  padding
+--------+--------+--------+--------+--------+--------+--------+--------+
|11011111|00000100|  data  |  data  |  data  |  data  |  xxx   |  xxx   | MAC
+--------+--------+--------+--------+--------+--------+--------+--------+
                  |<--------- MAC value ------------->|
```

[4] Optional Data Block (DB)

The DB has DL 8B-words, including optional padding (at the end) of PL bytes. Hence, the number of data bytes is 8*DL-PL. Both DL and PL are defined in the EEP-header.

The maximum length of the DB is 8*(2^25-1)B=256MB.

Type Extension (TE) 2 bytes

An extension of the following PT field. Secure PktWay uses the Type Extension field to contain a 16-bit key index. This key index is inserted by the Secure PktWay encryption unit, and indicates to the Secure PktWay decryption unit (at the destination SAN) which key should be used to decrypt the Secure PktWay packet.

Packet Type (PT) 2 bytes

The intent of the PT field is to provide all the information needed for demuxing in support of multiple protocol layers. In Secure PktWay the PT = Secure value specifies that the Data Block contains an encrypted Secure PktWay message, complete with another EEP header, and, potentially, prefixed L2-Routing-Headers.

Secure PktWay Secure Packet Type uses the preceeding 2 bytes of the Type Extension (TE) field for cryptographic key information.

The PT field will also indicate the commands used in the Secure PktWay Router to Router configuration and control Protocol (RRP).

Optional Header-Field Flag (H), 1 bit

This bit is set to 1 if there is one (or more) optional header fields following the standard 16-byte EEP-header. Since Secure PktWay makes extensive use of optional headers, it is expected that this bit will usually be set.

Source Address (SA) 24 bit

In PktWay this field is optional, and contains the address of the packet's original source in the same format as DA. In Secure PktWay, this field is mandatory, and contains the address of the Secure PktWay router which sent the message. This ensures that Secure Routers can use this field to identify the sender to which error messages may be returned

Part-2: Secure PktWay RRP messages

Secure PktWay is an open family of specifications for internetworking secure System Area Networks (SANs). This part describes the RRP (Router to Router Protocol) part of the Secure PktWay-protocol. It is built on top of the Secure PktWay-EEP described in Part-1. The packets of the RRP are listed, defined, and discussed in Part-3.

We introduce some new terminology within this document. A Secure PktWay Router always bridges (at least) two SANs. The Router consists of three parts: the "red" half router (HR) attached to the secure SAN, the "black" HR attached to the un-secure network, and their interconnection. The half routers which separate the secure SAN from the un-secure interconnection network are assumed to communicate through an encryption device. The operation of the encryption device is such that PktWay packets which are to be forwarded past the red half router to the black half router are first encrypted in their entirety, and encapsulated in an un-encrypted PktWay packet by the black half router.

Secure PktWay does not define the nature of the interconnection between the red and black half routers. However, the PCI Local Bus would be an ideal link interface. As mentioned earlier, Secure PktWay also does not define the interface or operation of the encryption device. Instead, it is assumed that this device is capable of rendering the Secure PktWay packet un-readable to the network adversary. It is also assumed that the encryption device will pass the key information, encryption parameters, and MAC’s needed for the Type Extension field, and the Optional Headers to the black half router. The black half router subsequently creates a new Secure PktWay Packet. The Packet Type is set to "Secure," the Type Extension field is filled with the key index from the encryption unit, the first Optional Header is filled with the MAC, the second Optional Header is filled with the encryption parameters, and the encrypted Secure PktWay packet is encapsulated in the Optional Data Block.

There are several implementation levels of Secure PktWay, for nodes and for routers. Although system designers are free to choose the level of implementation that best suits their needs, Redirect modes must be supported. This implies that the minimum implemenation level for Secure PktWay is Level-B.

RRP defines (via message structure and behavior) the interactions between HRs. RRP does not define the lower level protocols that deliver its messages (over links, or between processes in multi-homed routers). In particular, RRP does not define the inter-SAN interconnection links between the HRs that are left for mutual agreements among the implementors. These links are expected to range from serial fibers to PCI buses. A PPP-like protocol may be defined later for these links.

It is assumed that each HR has a Routing Table (RT) for its own SAN (aka Local Routing Table, LRT), with (at least) the addresses of all the nodes, and the source routes to each of them from the HR. This information could be dynamic or static, even manually configured. The HRs may (or may not) perform dynamic mapping of their SANs.

In L2 operation under levels C and D, when a source node, SA, needs to send a message to a destination node, DA, it first asks any of the HRs on its [SA’s] SAN for a source route (SR) from SA to DA. That HR would (1) provide such an SR, or (2) reply with a "Redirect" message, suggesting to ask another HR on the same SAN, or (3) report no knowledge of DA (using the UNK error message).

SA may ask more than one HR for the SR to the same DA and choose to use the best of these SRs.

In L3 operation, when a source node, SA, needs to send a message to a destination node, DA, it sends that message to any of the HRs on its SAN, using L2, expecting L3-forwarding to DA, using DA’s Secure PktWay address. That HR would either (1) forward the message toward DA, and possibly return to SA a "Redirect" message, suggesting to use, in the future, another HR on the same SAN for DA, or (2) report no knowledge of DA (using the UNK error message).

Part-3: Secure PktWay RRP Message Format

RRP messages are Secure PktWay messages with PT="RRP" in their EEP-header. The EEP-header is followed by some (zero or more) RRP-records according to their RRP-type, followed (always) by the MT which is the EI field.

The RRP-records constitute the DB of the Secure PktWay-message. They must be in Big-Endian order, with e=0 in the EEP-header.

The RRP-Type is carried in the TE of the of the EEP-header.

Following are the Secure PktWay RRP messages, with their RRP-type:

SECURE PktWay ADDITIONAL RRP MESSAGE SUBTYPES

RRP- Type	Impl'n Levels	Description
[GVSC]	BCD	Please negotiate a Security Context to node (address) The reply to [GVSC] is [IDSC] or [ERR].
[IDSC]	BCD	Here is the Security Context to node (address)

Secure PktWay RRP also uses the following error messages:

[ERR/PRIV]	BCD	Insufficient privilege for operation.
[ERR/SEC]	BCD	Incorrect security level.
[ERR/KEY]	BCD	Unrecognized, incorrect, or incompatible key.

All these messages may be sent from Secure PktWay nodes or HRs, to Secure PktWay nodes or HRs.

THE STRUCTURE OF THE RRP MESSAGES

The RRP-records are made of one or more 8B-words. In the following the RRP-type is in [] and its implementation level in (). Each message ends with an MT which is not shown here.

* [GVSC] (BCD) Please negotiate a Security Context to node (address)

```
        PH   (with [PT/TE]=[RRP/GVSC])
        ADDR (address of the node for which SCID is requested)
```

* [IDSC] (BCD) Here is the Security Context to node (address)

```
        PH   (with [PT/TE]=[RRP/IDSC])
        ADDR (address of the node for which SCID is provided)
```

SCID

RRP RECORD FORMAT

Each RRP-record starts with an 8B-word header as shown below. It first byte identifies the record type (RTyp). FILL IN RECORD FORMAT DESCRIPTION!

===> [SCID] Security Context Identifier Record

This record is used to negotiate a security context between two secure half routers. .

```
    0        1        2        3        4        5        6        7
+--------+--------+--------+--------+--------+--------+--------+--------+
| "SCID" | PL=0   |       RL=0      |Length1 |Length2 |Length3 |Length4 |
+--------+--------+--------+--------+--------+--------+--------+--------+
| Encr-T | Key-T  | MAC-T  |     Reserved    |   Security Context ID    |
+--------+--------+--------+--------+--------+--------+--------+--------+
| Encr-D | PL=0   |       RL=0      | Data1  | Data2  | Data3  | Data4  |
+--------+--------+--------+--------+--------+--------+--------+--------+
| Key-D  | PL=0   |       RL=0      | Data1  | Data2  | Data3  | Data4  |
+--------+--------+--------+--------+--------+--------+--------+--------+
| MAC-D  | PL=0   |       RL=0      | Data1  | Data2  | Data3  | Data4  |
+--------+--------+--------+--------+--------+--------+--------+--------+
| Sync-D | PL=0   |       RL=0      | Data1  | Data2  | Data3  | Data4  |
+--------+--------+--------+--------+--------+--------+--------+--------+
```

RRP MESSAGE EXAMPLES

==> [GVSC] Please negotiate an SCID from you to node-D (address)

```
    0        1        2        3        4        5        6        7
+--------+--------+--------+--------+--------+--------+--------+--------+
|00    P |         HR-Address         |     "GVSC"      |    "R R P"      |PH
+---+----+--------+--------+--------+-+------+--------+--------+--------+
|E=0|PL=0| Data-Length=1 (8B-words) |0|   0  |        S-Address         |
+---+----+--------+--------+--------+-+------+--------+--------+--------+
| "ADDR" | PL=0   |       RL=1      | AT=1   |        D-Address         |Addr
+--------+--------+--------+--------+--------+--------+--------+--------+
|     64 zero bits, unless any error was indicated along the path       |MT
+--------+--------+--------+--------+--------+--------+--------+--------+
```

==> [SCID] Here is an SCID to node-X (address)

```
    0        1        2        3        4        5        6        7
+--------+--------+--------+--------+--------+--------+--------+--------+
|00    P |          D-Address         |     "SCID"      |    "R R P"      |PH
+---+----+--------+--------+--------+-+------+--------+--------+--------+
|E=0|PL=0| Data-Length=8 (8B-words) |0|   0  |       HR-Address         |
+---+----+--------+--------+--------+-+------+--------+--------+--------+
| "ADDR" | PL=0   |       RL=8      | AT=1   |        X-Address         |Addr
+--------+--------+--------+--------+--------+--------+--------+--------+
| "SCID" | PL=0   |       RL=8      |               SCID                |SCID
+--------+--------+--------+--------+--------+--------+--------+--------+
|      64 zero bits, unless any error was indicated along the path      |MT
+--------+--------+--------+--------+--------+--------+--------+--------+
```

PktWay-WG <16> RRP-Format

```
==> [MLS?] Please tell me the security capabilities of Node-X
    (address | name | capabilities)
```

This message may have any of the following 3 forms:

If by Secure PktWay-address:

```
    0        1        2        3        4        5        6        7
+--------+--------+--------+--------+--------+--------+--------+--------+
|00    P |        HR-Address         |      "MLS?"     |    "R R P"      |PH
+---+----+--------+--------+--------+-+------+--------+--------+--------+
|E=0|PL=0| Data-Length=1 (8B-words) |0|   0  |        S-Address         |
+---+----+--------+--------+--------+-+------+--------+--------+--------+
| "ADDR" |  PL=0  |       RL=1      |  AT=1  |        X-Address         |Addr
+--------+--------+--------+--------+--------+--------+--------+--------+
|      64 zero bits, unless any error was indicated along the path      |MT
+--------+--------+--------+--------+--------+--------+--------+--------+
```

If by name (e.g., a name with 9 characters: A1...A9):

```
    0        1        2        3        4        5        6        7
+--------+--------+--------+--------+--------+--------+--------+--------+
|00    P |        HR-Address         |      "MLS?"     |    "R R P"      |PH
+---+----+--------+--------+--------+-+------+--------+--------+--------+
|E=0|PL=0| Data-Length=2 (8B-words) |0|   0  |        S-Address         |
+---+----+--------+--------+--------+-+------+--------+--------+--------+
| "NAME" |  PL=3  |       RL=2      |   A1   |   A2   |   A3   |   A4   |Name
+--------+--------+--------+--------+--------+--------+--------+--------+
|   A5   |   A6   |   A7   |   A8   |   A9   |  xxx   |  xxx   |  xxx   |
+--------+--------+--------+--------+--------+--------+--------+--------+
|      64 zero bits, unless any error was indicated along the path      |MT
+--------+--------+--------+--------+--------+--------+--------+--------+
```

If by capabilities (e.g., 2 capabilities, C1 with 2 parameter bytes, and C2 with no parameter bytes):

```
    0        1        2        3        4        5        6        7
+--------+--------+--------+--------+--------+--------+--------+--------+
|00    P |        HR-Address         |      "MLS?"     |    "R R P"      |PH
+---+----+--------+--------+--------+-+------+--------+--------+--------+
|E=0|PL=0| Data-Length=2 (8B-words) |0|   0  |        S-Address         |
+---+----+--------+--------+--------+-+------+--------+--------+--------+
| "CAPA" |  PL=1  |       RL=1      | CC=C1  |   P1   |   P2   |  xxx   |cap
+--------+--------+--------+--------+--------+--------+--------+--------+
| "CAPA" |  PL=3  |       RL=1      | CC=C2  |  xxx   |  xxx   |  xxx   |cap
+--------+--------+--------+--------+--------+--------+--------+--------+
|      64 zero bits, unless any error was indicated along the path      |MT
+--------+--------+--------+--------+--------+--------+--------+--------+
```

RRP-Format <17> PktWay-WG

```
==> [ERR/PRIV] Insufficient Privilege
```

```
     0        1        2        3        4        5        6        7
+--------+--------+--------+--------+--------+--------+--------+--------+
|00   P  |          D-Address        |      PRIV       |    "E R R"      |PH
+---+----+--------+--------+--------+-+------+--------+--------+--------+
|E=0|PL=0| Data-Length=? (8B-words) |0|   0  |          S-Address        |
+---+----+--------+--------+--------+-+------+--------+--------+--------+
|                                                                       |Data
|<------The entire message that could not be handled by the sender----->|Data
|                                                                       |Data
+--------+--------+--------+--------+--------+--------+--------+--------+
|      64 zero bits, unless any error was indicated along the path      |MT
+--------+--------+--------+--------+--------+--------+--------+--------+
```

This message reports that the enclosed message could not be handled by its receiver (the sender of this error message).

Appendix-A: Enumerations

(A1) Secure PktWay Packet Types

The EEP header reserves 4 bytes for signaling from the source node directly to the destination node. They are the PACKET TYPE (PT), and the TYPE EXTENSION (TE), 2 bytes each.

This list defines additional values for the PACKET-TYPE (PT) field. Each packet-type has its own interpretation of the TE and the F fields.

Code	Packet Type	Type Extension	F-Bit
121	Secure	Key Index	1

This implies that a PT="SECURE" will use the TE field for a 16-bit key index, and that the F-bit is '1' -- indicating that Optional Headers will be associated with this Packet Type.

(A2) RRP Messages

RRP- Type	Code	Description
GVSC	41	Please negotiate a Security Context to node (address)
IDSC	42	Here is the SCID to node (address)
MLS?	43	Please tell me the security capabilities of node (address | name | capabilities)

Throughout this document the RRP messages are indicated by their type (e.g., RDRC for Redirect). In actual messages the code is used (e.g., 2 for RDRC).

(A3) RRP Records

RTyp Code Description

SCID	55	Security Context Record

(A4) Error Messages

Subtype	Code	Description
PRIV	101	Insufficient privilege for operation
SEC	102	Incorrect security level
KEY	103	Unrecognized, incorrect, or incompatible key

Throughout this document the error messages are indicated by their subtype (e.g., PRIV for Privelege Error). In actual messages the code is used (e.g., 5 for PRIV).

(A5) Secure PktWay Node Capabilities

Code	Capability	Parameters
254	Security	Encryption Parameters

(A6) Additional Optional Header Fields Types

The MSbit of the OH-type-field is the type-of-type. The assignment is:

0b0: Optional (may drop this OH if its type, tttttt, is unknown)
0b1: Mandatory (should not process this message, if it's unknown)

The next bit is the "Completion bit" (C). Its assignment is:

0b0: More options follow
0b1: This is the last option field

The 6 LSbits are the type field. Their assignment is:

0x06: MAC here
0x07: MAC following the DB
0x3E: Cryptographic Data

Appendix-B: Example of the SCID Negotiation and Communication

The following is an example Secure PktWay scenario, involving static keys and dynamic Security Context Identifier negotiation.

I. SETUP

1. Keys are loaded into each encryption unit at boot time. Each encryption unit contains keys that it uses to communicate with every other encryption (e.g., the encryption unit for SAN X contains keys for SAN X - SAN Y, SAN X - SAN Z, etc.).
   Each red HR is given permission to assign some number of key indices to Security Context Identifiers of its choosing. Each black HR is given a table which maps a key index to a destination black HR address.

2. Each red HR maintains a mapping of Security Context ID to key index. The SCID format is arbitrary, but let's assume that it contains all or part of the black HR address of the originating SAN, so the red HR can assign SCIDs without worries of conflicts with other HRs.

3. Node 1 on SAN-A (A1) wants to set up a security context with Node 2 on SAN-B (B2). It sends a message to SAN-A's red HR (rA), asking for a new SCID to be negotiated. rA creates a new SCID, associates it with a currently unused key index. The red HR then negotiates the SCID with the red HR at the destination SAN so the SCID's may be correlated on return traffic. The local red HR then returns the negotiated SCID to A1.

II. MESSAGE

1. A1 generates a message for B2. The message will use L3 forwarding, so it includes the regular EEP header with B2 as destination, preceded by a symbol which contains the SCID for this message. A1 routes the message to rA via native SAN routing.

2. rA receives the message and retrieves the SCID from the table. It retrieves the key index associated with the SCID and gives the following to the encryption module: key index, message length (including symbol), and message (including symbol).

3. The encryption unit at SAN A encrypts the entire packet, including the symbol, and gives the following to the black HR (bA): key index, encryption parameters, encrypted message length, encrypted message, EEP MAC, and ODB MAC.

4. bA looks up the destination index and address and creates a PktWay packet with PT=SECURE. The packet is routed via PktWay L3 routing to the destination black HR (bB).

5. bB receives the message and gives the following to the decryption module: key index, crypto parameters, encrypted message length, encrypted message, MAC.

6. Crypto B decrypts the message and gives the following to the red HR (rB): key index, message, and some indication whether the authentication succeeds or fails.

7. rB adds mapping from incoming SCID to key index and routes

message to B2.

8. B2 saves the SCID (if necessary) to use for any reply messages that it may generate for A1. (It might use a different SCID instead.)

Appendix-C: Example of the use of Secure PktWay RRP

[Blank]

Appendix-D: Glossary

Covert Channel: A communication channel that allows two or more collaborating processes to transfer information in a manner that violates the system's security policy.

Key Management: The generation, storage, distribution, deletion, archiving, and application of cryptographic keys in accordance with a security policy.

Secure Router: A router which provides security services, e.g. cryptographics functions, for trusted clients when they communicate via external untrusted networks.

Security Context: A relationship between two or more entities that describes how the entities will utilize security services to communicate securely.

Threat: A vulnerability available to a motivated and capable adversary.

Trusted: The belief that a system meets its specifications, and in data security, pertaining to hardware and software systems that have been designed and verified to avoid compromising, corrupting, or denying sensitive information.

[Blank]

Appendix-E: Acronyms and Abbreviations

GVSC	An RRP message, requesting the negotiation of a SCID
KEY	An error message, indicating an incompatible key
MAC	Message Authentication Code
MLS?	An RRP message, requesting security capabilities
PRIV	An error message, indicating insufficient privilege
SCID	Security Context Identifier
SEC	An error message, indicating incorrect security level

[end]

SANDERS
A Lockheed Martin Company

Evaluating SHARE SAN Communication

(HPSC SHARE IPng(V6) and Packetway)

Sanders, a Lockheed Martin Co.
Advanced Technologies
P.O. Box 868, PTP2-D001
Nashua, NH 03061-0868

1. Abstract

The secure heterogeneous application runtime environment (SHARE) program at Sanders depends on its ability to exhibit beneficial, unique, value added capabilities. Particular to SHARE is the ability to coexist with non-SHARE components in a high performance networked environment. This, in part, is accomplished by providing security while maintaining high performance scalable communications. This document examines the use, needs and methods of communication models best exercised in a high performance scalable computing (HPSC) SHARE. It focuses on two protocols in particular, IPng (v6) and Packetway. These protocols are compared for their ability to deliver required SHARE system area network (SAN) communications while allowing the ability to communicate effectively with conventional networks.

2. Terminology

2.1 System Area Network (SAN)

A SAN is an opaque high-performance computing unit that may contain many computing components. Among these components are nodes, composed of processes or processing elements. Physically, a system is considered a SAN if at some point the processing within the indiscernible unit passes through a component that exports or imports information on behalf of the collective. Unlike a WAN or LAN, a nominal performance network, where the division is made based on the size or proximity of a subnet, a SAN specifically addresses an application or subsystem application where tightly coupled communication or parallel operation produces information representative of the collective computing elements. Additionally, a SAN is said to function at a single security level where a LAN or WAN may employ multiple. A SAN therefore, is a tightly bound system of elements which do not require the need of router or bridge logic (layer 3 routing) to inter communicate or operate. Furthermore, a SAN may use a homogeneous layer 2 protocol, potentially proprietary in nature, and most likely switched based if (point to point) performance is a consideration.

2.2 SHARE communication

SHARE builds upon the High Performance Scalable Computing (HPSC) program at Sanders. To this end, SHARE must remain somewhat high performance and scalable while endeavoring to provide security. SHARE also contends with the problem of remaining operable with non-secure SANs or WANs that may be lower performance computing domains. Simply stated, SHARE must communicate at rates required by all network interconnects while maintaining multiple levels of security. Although the majority of SHARE communication is intraSAN, this area is treated as opaque. However, because of the SAN communication boundary, SHARE is stressed to deliver performance, security and connectivity. Consequently, SHARE communication focuses primarily on the interSAN and lower performance networks. (As alluded to, this exposes a weakness to which SHARE poses a solution. However,-SHARE interSAN communication establishes the foundation for this study.)

2.3 High Performance

The SHARE model requires high performance, massively parallel, distributed applications, the kind of transparent data communication to which they are designed or accustomed. High performance however is a subjective term. SHARE views high performance as the ability to deliver or acquire information at the rate necessary to maintain application performance without interruption or distortion. SHARE further stipulates that while using Myrinet, a minimum of 160MBs full duplex must be maintained. High performance also connotes point-to-point, message passing, real-time determinism, voluminous data movement and correctness.

2.4 SHARE Router

A SHARE router is an intelligent network component capable of learning, performing independent peer to peer dialogue and maintaining multiple security policies simultaneously. A SHARE router is also capable of bridging the SAN to LAN or WAN boundary that may require the conversion of packet or information format. A SHARE router is an independent SAN, but most importantly, with unique attributes allowing it to appear as a component of the SANs or conventional networks with which it is connected.

2.5 InterSAN

InterSAN communication is information transfer across SAN boundaries requiring a router. The SAN boundary is determined by three factors, proximity, algorithm or protocol. Physical separation or distance between communicating peers may logically dictate a SAN. Similarly, partitioning of an algorithm constitutes a SAN distinction. More overtly a difference in native protocols identify a SAN boundary. In all cases, a SHARE router is required. The strongest case for a router exists as a result of a disparity of network protocols and/or security levels. Where there are differences in security levels or native communications a SHARE router is required to adjoin the transient information with encryption or translation information. The result in all cases is that the communications within the SAN becomes opaque and can be treated as a black box. The important distinction is that interSAN communication, unlike LAN or WAN communication, which is eclectic, is algorithm centric whether at a distance, different security level or native speaking network.

2.6 IntraSAN

IntraSAN communication deals specifically with data conveyance within the SAN boundary. Unlike a LAN, intraSAN connotes high performance, scalable, tightly coupled, application centric, information exchange. Similarly, SAN nodes function in a cohesive, homogeneous manner and use at most a native layer 2 protocol. Although intraSAN communication is beyond the scope of SHARE, it is worth noting that the protocol choice made by the SHARE at the interSAN may drastically affect the appearance, performance and operation of the components of the SAN, and thereby alter the SAN boundaries. This may prove to be an undesirable a side effect. The consequences may require the rework of algorithms, that may be timing sensitive, with the intervention of a router.

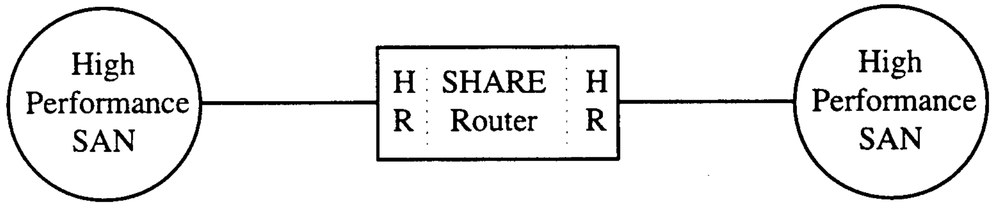
3. Approach

The approach of this paper, analyzing the SHARE communication models, is qualitative in nature and expands from the simple to complex case. The following subsections present the basic communication problems considered critical to the success of the SHARE. The proposed common infrastructure protocols are examined in light of these scenarios and compared for their ability to meet the specified criteria.

3.1 Problem Set 1

The first form of communication examined is that of interSAN (Figure 3-1). This is the simplest case, because it involves only SANs and the SHARE router. Each SAN focuses solely on high performance, real time algorithm computation. This case is homogeneous in that any heterogeneity is handled within the router. This case presents a deterministic environment at the router interfaces isolating the router and any protocol or security translation. Consequently, this scenario is easily analyzed. What makes this problem interesting is that the choice of protocol (should both perform all required SHARE criteria) by header content, arrangement and interpolation exacerbate(s) the determinism and consequently the performance of the respective SANs. Hence, the use of Internet Protocol Next Generation Version 6 or the contending Packetway clearly identify a preferred approach.

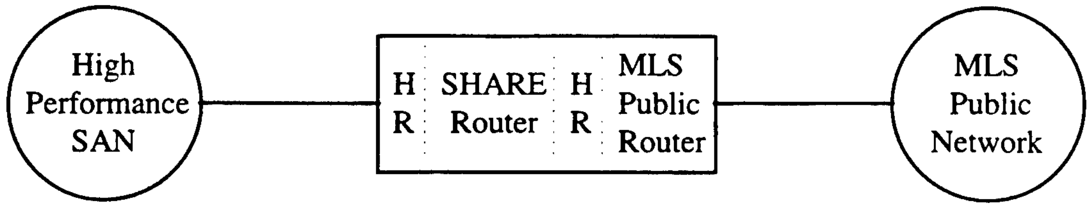
Figure 3-1 Case 1: InterSAN Communication



3.2 Problem Set 2

Problem set two is slightly more complex (Figure 3-2). One of the high performance SANs is replaced by a non-SHARE (conventional) LAN or WAN. Unlike SANs which have a proximity, security/protocol and algorithm centric philosophy, the LAN or WAN limits only the proximity. Constrained by communications, not the network connections, data transfers are indiscriminate, variable in format, scope, security levels and complexity and hence place additional translation burden on the connecting router. Furthermore, the protocol essentially becomes heavier weight to handle the unpredictable nature of the information exchanges. Similarly, the intervening routers must also increase in intelligence, memory usage, transmission control and complexity. Moreover, router configuration requires protocol conversion interfaces that allow SHARE SAN and conventional Internet traffic exchange. Although high performance is less likely an issue, the ability to communicate with a graphic user interface, or rendering engine is. Hence, a ubiquitous protocol to all networked elements is essential.

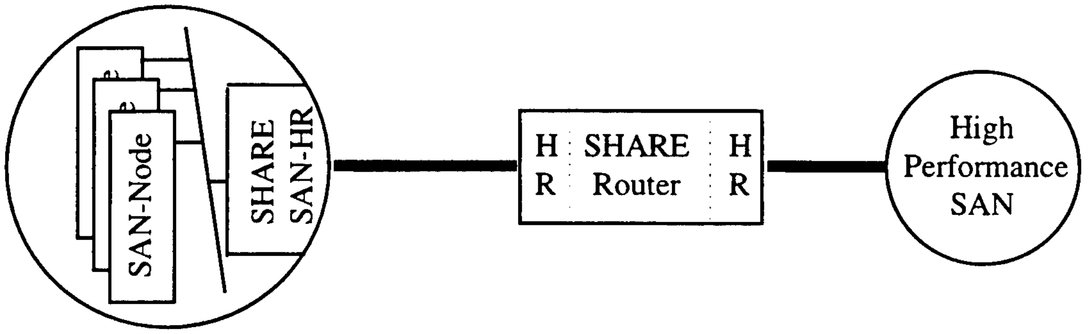
Figure 3-2 Case 2: SHARE SAN and Public Network Communication



3.3 Problem Set 3

InterSAN and half SHARE half conventional communications comprise a proportionally small amount of the application data transfer when considering all communications within a SHARE domain. (Figure 3-3). Moreover, of the first two problem sets, interSAN communication is the greater. In actuality, the majority of communications resides within the SAN. In addition, when this information must be conveyed elsewhere (interSAN) additional data is most likely being queued, ready for transmission. If the interSAN communication protocol proves too slow, cumbersome or inadequate, the intraSAN application suffers. It would make sense then to consider the most critical and voluminous data transfers in the evaluation and selection of an interSAN protocol. Another consideration would be how the data is to be presented or used. If the application generates imaging data, then the user interface may function at a rate significantly less than that of the data generators. In other words, what the application does and how it behaves has an impact on the communication model that is used at all locations. It may be that two distinct levels of protocol are employed. One within a SAN, which is lightweight and efficient, but a heavier weight protocol may be perfectly acceptable interSAN or between SANs and conventional networks. In any case, the solution to the protocol selection is best examined from the inside (intraSAN) out (interSAN).

Figure 3-3 Case 3: Intra and inter SHARE SAN communication



4. SHARE Problem Constraints

InterSAN communication is the principle focus of the SHARE communication domain. However, as stated earlier intraSAN communication is a significant consideration in the evaluation and adoption of the SHARE interSAN protocol. Furthermore, the selection of a SHARE protocol must include flow control and adroitness in the requirements imposed by both types of communication. Similarly, as SHARE implementations include conventional local and wide area networks, the selection of a SHARE protocol must include consideration of multiple criteria; most notably, reliability, security, distributed intelligence, voluminous data, message passing, heterogeneity and timeliness. Clearly, in view of these diverse demands, the desired protocol needs to flexible, extensible, scalable and robust, and yet, light weight. In other words, the protocol or protocols selected by the SHARE must function transparently, efficiently and effectively for the purpose of appropriately secure information exchange.

4.1 Security

The SHARE security requires, with little or no performance compromise, the encryption and decryption of all packets through routers. SHARE routers, consequently, are equipped with special hardware to encrypt and decrypt. The protocol must facilitate security as part of its rudimentary design while allowing all exchanges regardless of security level, association or transport methodology. Simply stated, the protocol must innately specify or make provisions for security.

4.2 Reliability

The underlying protocol though not necessarily providing connection reliability must permit the applications the ability to do so at no additional cost. That is, applications, which employ connection or connectionless communication, shall not be impacted or indisposed as a result of the protocol. Furthermore, the protocol may not add indeterminate latency as a result fo the transport methodology employed. The protocol selected by SHARE must maintain a consistence behavior at each of the junction points, intraSAN, interSAN and SAN/non-SAN. If a single protocol can not provide such capability, then multiple protocols may need to be employed to express the transfers required by SHARE applications.

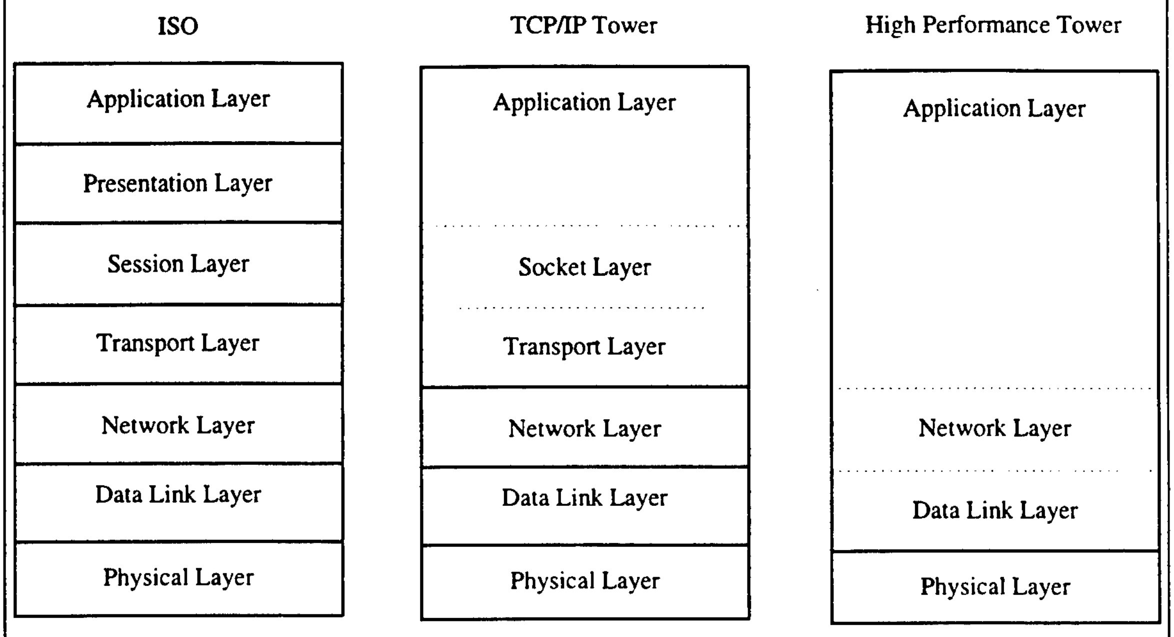
4.3 Voluminous transfers

The SHARE protocol must provide the capability to multiplex information quickly to best serve the needs of the algorithm, distributed process, or routing. The overhead to handle data transfers must incur low latency. Correspondingly, the protocol header must be easy to parse, simple in design and yet adroit in terms of information conveyed. The same should be true regardless of the packet size or content. Therefore, the protocol, the header itself, must be significantly low in non-data transfer related processing. In high performance computing, high data rates are expected with numerous transfers occurring simultaneously. This is especially true within a SAN, at the heart of the algorithm execution. Such networks tend to be switched-based and, therefore employ cut-through routing. Hence, the protocol must support cut through routing capability as well the conventional store and forward methodology.

5. Protocol Adjudication

Today, a plethora of protocols is available within the various levels of the common network towers (Figure 5-1). Each protocol has its strengths and weaknesses. Typically, a protocol is required whenever more than one type or format of information needs to be communicated among aggregate networked entities, whether they are physical (processor) or abstract (nodes). Accordingly, protocols have been designed to address specific control needs for a variety of systems. As such, the system developer has the responsibility of determining whether one or more protocols are required to control and facilitate network communications. Moreover, many other criteria must be considered once a protocol need has been established. These criteria are system dependent but share a common base domain of factors.

Figure 5-1 Commonly Accepted Network Towers



- Quality of Service
    - Rate
    - Reliability
    - Type of Service
- Heterogeneity
    - Hardware
    - Application
- Cost
    - Monetary
    - Risk
- Development
    - Program Management
    - Application programming interface
- Support

    - Maintenance
    - Endemic with similar systems or issues
- Longevity
    - Survival
    - Extensible

The following sections explore some of the thought processes when evaluating a protocol for use, particularly in embedded systems.

5.1 Proper Protocol

In many cases, system architects, using the common factors itemized above, determine that no commercial protocol serves their specific needs. They, therefore, devise unique point solutions, often in the form of a proprietary protocol. Ultimately, time determines the correctness of the decision.

Today, many system architects wish to employ open standards and, therefore, seek protocols that are a best fit rather than perfect fit. Significant work is being done conforming old protocols and to develop new ones for these emerging technologies. As an example, applications are tending toward massively parallel programming (MPP) and extreme performance. Of importance to such programs is service information content without the cost of latency. Usually, it is latency that causes protocols to be discarded, especially where the data transfer size is small and the protocol header is large. The principal dilemma then, when selecting a protocol is that of striking this appropriate balance between countervailing performance attributes, information content and latency. Information content and other services increase latency or decrease communication speed, as these are processes, which delay actual data transfer. Delay or latency is directly manifested in a relatively large protocol header as the inclusion of service content may be bulky. Protocols that are biased on the side of quality of service are likely to be discarded whenever the implementing protocol header is large, on average, relative to the packet payload. In other words, protocols are not likely to be considered, if the process delays associated with available services are large relative to the time associated with actual data transfer (core) processing. This is certainly true for embedded systems where very fast communication is essential, especially intraSAN. However, the further communication is from the core algorithm processing, the more likely existing protocols may handle the transfer volume. This would be true for GUI interfaces presenting image data. Since the refresh rate of a screen rarely exceeds 70Hz, faster data arrival would be inconsequential. Consequently, a last in first out queuing model and any level 3 protocol would be sufficient. Moreover, a protocol with a larger header becomes of less importance. Again, the further out from the core algorithm processing, the more the focus is on the services the protocol can provide rather than its speed.

However, protocol considerations are greater than simply an assessment of latency based upon the relative size of the packet payload versus the protocol service information content. Firstly, although a protocol header may be large, i.e., may provide many fields for the selection of services or functions, not all of these must be selected. Some functions may be out of scope for a given system as is often the case for embedded systems. Non-essential or service related processing time, therefore, can be minimized by carefully weighing the needs verses the services. Note, however, that all services, selected or not, by virtue of associated header fields, contribute to processing delay as all header fields must be parsed. Secondly, performance also can be impacted by the relative proximity, size and alignment of the fields. The recent trend toward the employment of hardware accelerators has mitigated the delay. However, not all systems employ such devices thereby alleviating alignment of odd sized fields that would otherwise have to be parsed and interpreted by software.. Consequently, most prominent and enduring protocols make use of fields aligned on octet or word boundaries and only use nibbles for fields conveying information of limited value. The bottom line is, as new algorithms and applications are developed, the protocol selected has to be able to conform, integrate and serve the communications needs. The protocol must do this with services, reliability and low latency.

5.2 Rethinking the Embedded Network Tower

In today's embedded systems, the tower is shrinking. Whereas once the approach was to modularize each network function, today, collapsing tower layers, discarding some function and streamlining the processing is prevalent. The result is protocols that are sensitive to the needs of the embedded industry. These protocols have the look and feel of low level protocols but provide the capability of supporting the higher functions required by intelligent system network controllers. The network controllers are getting smarter too.

Today, hardware accelerators, in the form of an FPGA, are being employed more often. The benefit acquired from allowing a low-level protocol to convey high level function with the aid of an accelerator greatly enhances the processing capability of the overall system and network. Though this is the trend, not all embedded systems are there. It is therefore still critical to develop or use a protocol that is conducive to an accelerator but also easily parsed by software. Collapsing the protocol tower may have devastating effects on protocol headers as a result of compacting information. How the information is arranged, conveyed and deciphered becomes critical in each layer.

While academia is collapsing the network tower, others are adding new layers. These layers are smaller and focused for specific purposes. Some layers are added and removed as needed. This type of thinking is referred to active networks, a growing trend among embedded system developers. In active networking, where memory and time is limited, functions are loaded and unloaded. Should some function be requested, that capability may be solicited by the network interface controller, loaded, and executed. The result is added capability, but the burden placed on the protocol is severe. The protocol must be able to accommodate such diverse and dynamic procedures. To handle this, developing protocols are being prefixed with symbols or tags. The protocol, therefore, becomes an active carrier of the network. As a result, the network, its protocol and processes are continually in a state of flux. This thinking promises increased throughput, lower latency and more services because the protocol and layers together shape the information transmitted and received. Unfortunately, the network science of this nature is still immature.

Whether adding or removing layers, using one or more protocols, the goal is to support the applications. If the network connectivity meets the criteria of the application, the system goals may be attained. In conclusion, the idea is that a network and its accompanying protocol(s) must be transparent, efficient and reliable.

5.3 Hardware Acceleration

The idea behind a hardware accelerator is simple. If the packet format is well defined, hardware may be enabled to perform decision processing on the packet header stream as it arrives. Increasingly, commercial routing vendors are using hardware accelerators at various layers to achieve increased delivery performance. One such vendor, Cisco (7513 series routers) has gone as far as to employ FPGA technology to reconfigure the hardware acceleration based on the arriving protocol. The performance enhancement provided by the hardware alleviates the burden of many lines of assembly code to perform the most mundane decision processing associated with the reception of a packet.

5.4 Header Format Ordering Increases Performance

The format of the header fields is important when high performance is required. Best performance is achieved when decisions about the packet can be made while it has not fully arrived. One such model which uses the format of the header is cut-through routing. The idea with cut-through routing is that the entire packet need not be resident in memory prior to delivery or commencement of delivery to another or ultimate destination. The format of the header then becomes critical in determining the meaning and purpose of the packet as the header words arrive. The development or re-engineering of most protocols include consideration of these facts.

Protocol format decisions are not limited to the header alone. In some instances, the format of the entire packet is considered to address such needs as security. The format of the header is critical in determining how to handle the data, but the header alone may not be independent of the trailing packet, including the tail. This is specifically true if the security measure employed is encryption. How the header is arranged may limit the ability to encrypt and/or expose information unnecessarily.

5.5 Weight and Layering

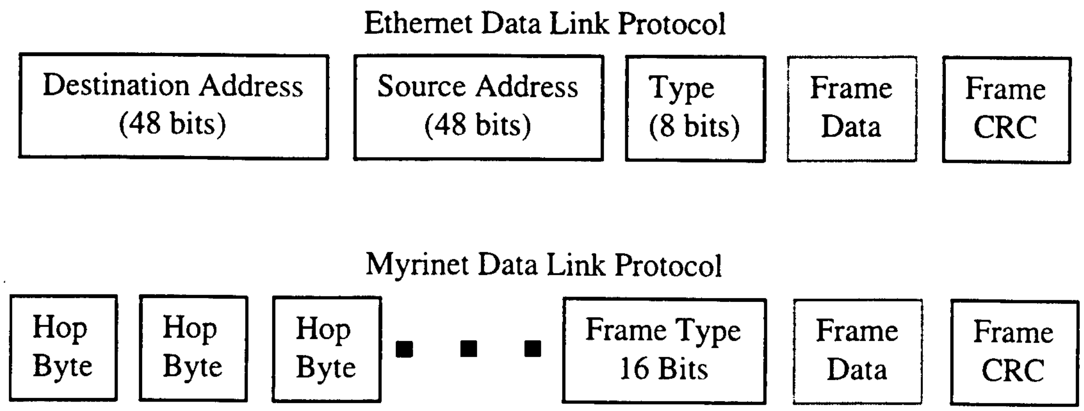
A protocol is said to be heavy weight when it contains information not necessary or counter to the immediate goal of conveying or expeditiously deciphering a course of action. Of course, it must be stated that the weight of a protocol reflects the totality of usefulness. This means that the information conveyed in the header may be meant to accommodate a variety of purposes. Consequently, no one operation is performed more efficiently than another is. Similarly, few operations are excluded entirely. This is most commonly true of protocols as they make their way up the traditional network tower. Where the lower protocols perform a single task; the higher protocols are called upon to be increasingly intelligent. This intelligence ultimately leads to a lack of performance and focus. The resulting protocol then is made to carry or incorporate pieces of information that would better be suited to some newly devised intermediate layer. A perception of some advocates of layer reduction or streamlining is that significant and unnecessary latency is entailed in the parsing and demultiplexing operations associated with unnecessary layers. This fallacy has caused even a greater shift to collapse intervening layers. This is not to imply that some collapsing of layers is inappropriate, but that separating layers should not be viewed as the nemesis of performance. Studies have shown, to the contrary, that the proper separation of information amongst layers leads to increased performance, modularity of design and extensibility.

5.5.1 Data Link Protocol

The Data link protocol is the lowest software protocol. The data link layer provides nothing more than point to point connectivity and a mechanism for services. In Figure 5-2, the two data links, Ethernet and Myrinet, illustrate this point clearly. Note that both protocols provide only encapsulation for transfer of frame data to a designated destination. Notice, Ethernet uses a forty-eight bit physical frame address to identify the destination. The data link packet remains intact all the way to the destination. At the destination, the frame data is directed to a service access point based on the information identified in the eight-bit service type field.

In switched-based networks, which are more of the high performance variety, the data link works slightly different. The data link header is stripped with each hop through switches. In the case of Myrinet™ by Myricom, each hop removes an eight-bit octet. The final element to receive the frame evaluates what action is to be taken. The action is determined by evaluating the immediate sixteen-bit frame word that is the service access value. In either case, Ethernet or Myrinet™, no work other than handing the packet of data to a service access point is performed at the data link layer.

Figure 5-2 Example Data Link Protocols



5.5.2 Network Layer Protocols

The network layer is where the real work for maintaining connectivity resides. As such, network protocols must be able to accommodate numerous services including such functions as, fragmentation, reassembly, destination routing, source routing, network accessibility, scouting, time to live and multiplexing for higher levels. Unfortunately, this accommodation produces heavy weight headers with significant parsing and processing time. The trade off is for work that may be done verses the timeliness of delivery. Most network protocol implementations only do a subset of the possible network layer services (e.g., IPv6). Principally, this is because system architects must economize on their selection of services in order to address needs like performance requirements and system constraints. Embedded systems, for example, typically have a memory constraint that mandates selection of only a few essential services. Unfortunately, reducing the number and types of services does not yield a proportional decrease in time. Similarly, it does not result in a decrease in the size of the protocol header.

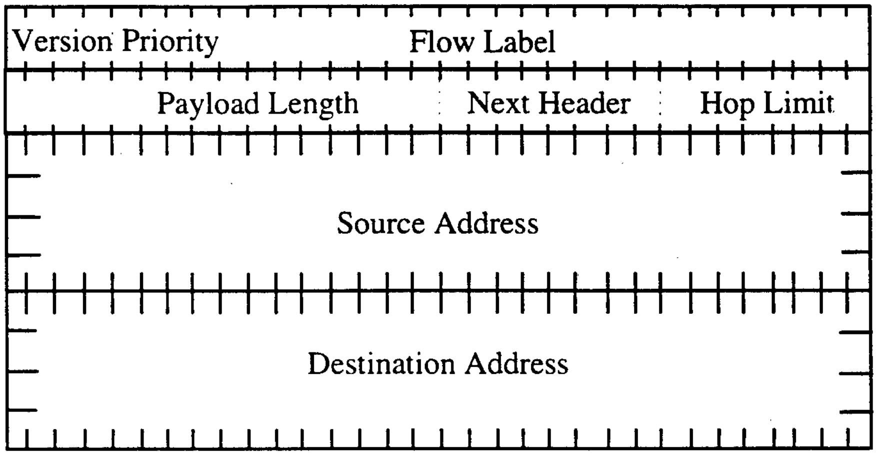
6. Internet Protocol Next Generation (Version 6)

In 1982 the Internet Protocol began to experience growing pains which provoked the Internet Engineering Task Force to evaluate the future of Internet Protocol communications. It would not be until December 1995 that the IETF would ratify RFC 1883, Internet Protocol, Version 6 (IPV6) Specification. This new protocol specification was an outgrowth of the continuing effort to develop the next generation solution. Internet Protocol, Version 6 is the product of talented software, network and hardware engineers who comprise the Internet Protocol Next Generation Working Group at the IETF. Collectively, many years of experience are being brought to bear on the solution to the next generation Internet Protocol and the types of communication it must support.

6.1 Ipv6 Header

Internet Protocol version 6 makes many provisions for future Internet communication, including high performance and telecommunications. The newer protocol rendition has fit more processing information into the first two thirty-two bit words than its version 4 progenitor. Following the first two thirty-two bit words is the destination and source addresses that have been expanded to one hundred twenty eight bits for future addressing accommodations.

Figure 6-1 Internet Protocol Version 6 Header Format



The information layout provides for the quickest decision making and demultiplexing ability of any previous Internet Protocol. Now with IPv6, decision processing can be done early, even as the packet is arriving. The protocol supports cut through routing capability as well as the traditional IP store and forward paradigm. The intent is to increase the overall performance, remove unnecessary overhead and streamline its implementation. Furthermore, IPv6 has accommodations for much larger packets, hence offsetting proportionally the increased physical header size. Most importantly, IPv6 introduces the flow label and next header field.

The modifications to the twenty-year-old header permit streaming and immediate demultiplexing respectively for the newest of network controllers. The Internet Protocol Version 6 header (Figure 6-1) is compact (few words) and succinct (few fields) and already purports a growing audience of participants. Primary among the users are router or gateway vendors whose newer products are making extensive use of the robust features. Of particular note is Cisco, with its 7500 series routers. Cisco publishes 2.1 Gbs throughput using cut through routing, high availability and support for traditional as well as telecommunications networks.

When considering this protocol for high performance networked applications, what is immediately obvious is that there really is not too much to this header outside of two thirty-two bit words. All the fields are a minimum of a nibble where nibbles adjoin to keep byte and word alignment. The fields are arranged to allow maximum information knowledge as the packet arrives from the network. Whether this protocol is deciphered in software or assisted by a hardware accelerator, it is simple, succinct and robust. This protocol header gives good indication that it is well arranged and capable of handling either end point or router to router information efficiently.

6.1.1 Version

The version is a four-bit field. It is the first nibble of the first word received by a network interface controller. Given a hardware accelerator or a modular software implementation, immediate formatting or parsing methods can be invoked. With four bits identifying the version, this particular header can accommodate fifteen renditions, but all variants are specifically variants of the header type, IP.

6.1.2 Priority

Priority allows the lowest layer the ability to apply special handling such as queuing or security. Four bits identify the field that can accommodate fifteen types of priority. The priority field completes the first byte of the two-word header.

6.1.3 Flow label

The flow label, a twenty-four-bit field, may be used by a source wishing special handling of the packet by IPv6 routers, such as non-default quality of service or real time service. It may establish streams of communication as cached flow handling state information. Primarily for accelerating routing, it may also be used by end point elements to expedite voluminous data flows of a known or prearranged method.

6.1.4 Payload Length

The payload length field is a sixteen-bit field and identifies the number of following octets. The limit of an IPv6 packet therefore is 65536 octets which makes the header $1.2X10^{-2}$ percent of the total packet size; Packets however can be changed. The length field being in the first word makes it easy to invoke or set a hardware accelerator (DMA) to move the packet off the wire.

6.1.5 Next Header

The next header is the most valuable addition to the packet header. It is an eight-bit selector. It identifies the type of header immediately following the IPv6 header. This is repeated until the next header signifies no further next header exists. As a result of the adoption of the next header field, IPv6 enables infinite extensibility and accommodations for such concepts as active IP networks; a new venture in network intelligence.

6.1.6 Hop Limit

This field is the last eight bit unsigned integer of the second header word. This field is decreased each time a node forwards the packet. The packet is discarded if the hop limit decrements to zero.

6.1.7 Source Address

The source address field, one hundred twenty eight bits of hierarchical address resolution, identifies the originator of the packet. This field is always present in the header that permits a first level of authentication for a receiving node. It is also used by routers to "listen" for nodes. The source address traditionally has provided the numerous other IP related protocols the ability to keep network state information, routing tables, hop limits and the like. In other words, beyond identifying the source, the source address is the catalyst for a plethora of additional operations.

6.1.8 Destination Address

The destination address field contains one hundred twenty-eight bits for hierarchical address resolution. It may address a physical destination or a logical destination. In either case, the field is always the last field of the IP header. If a routing header is present, the destination address may not be the final hop (see RFC 1884) and section 4.4 of RFC 1883.

6.2 IPv6 High Performance Computing Advantages

The IPv6 protocol comes with a host of engineering heritage. Every detail of the IPv6 header format has been given significant thought by numerous network engineers. The header format has been changed to increase address space, enable cut-through routing, facilitate high performance computing and provide support for highly distributed applications. To this end, IPv6 has attained its goal of addressing growth while increasing performance. Furthermore, IPv6 brings much in the way of conformance and stability to higher performance computing by making provisions for hardware accelerators and new software interconnect innovations.

6.2.1 IPv6 Hardware Acceleration

The IPv6 packet header is easily configured for hardware acceleration. There are eight fields; each aligned on nibble boundaries. The fields are ordered to allow cascaded decision processing. Additionally, with the introduction of the flow labels, hardware can be designed to specifically handle streams of data efficiently. Flow labels primarily allow the setting of end point connections and then permit the streaming of continuous data point-to-point. For the life of the flow label, there would be no further need of software intervention.

Hardware accelerators can also be used to sort according to next header information. That is, by programming the hardware interface to accommodate known next header layering. The packets can be internally routed to destinations suited to handle them.

6.2.2 IPv6 Infinitely Scalable

The IPv6 committee took special care to design for the infinitely scalable network. The source and destination address fields were expanded to one hundred twenty-eight bits for just this reason. Furthermore, the Internet protocol has already demonstrated its ability to scale network connectivity and as a result to distribute application workload. A prime example of expanding distributed processing is the wide spread invocation of server type applications, especially in the realm of data basing, image processing and telecommunications.

6.2.3 IPV6 Cut Through Capability

The IPv6 header has been developed to take advantage of cut through routing. The format of the IPv6 header fields have been configured to facilitate the efficient parsing of packets as they arrive. The first thirty-two bit header word format presents a hierarchical methodology which permits effective decision tree processing, starting with the version and continuing through to the extended addressing fields. IPv6 implementations may exploit theses fields for specific purposes which increase the throughput or serve the particular purpose of the network device.

The second thirty-two bit word, are used when additional information regarding the packet and its purpose are of interest. Of these fields, the next header field is the most interesting. Since an IPv6 packet can encapsulate multiple other types of packets and information, the next header allows the network layer the ability to decipher or glean more information from the packet and hence apply particular algorithms as needed.

The two remaining fields, the source and destination addresses, are placed last because they would most likely be used by intermediate nodes, usually routers. Routers are usually slower than end point nodes and perform network-related operations rather than application specific functions. Thus, software acts on the whole packet header and require information about the source and destination. Conversely, an end node would most likely skip these fields once a stream or connection has been established.

6.2.4 Popularity

IP is a very popular protocol. As a level three protocol, it comes with a significant amount of header information content that allows a multitude of level 3 functional capability. As far as level 3 protocols go, IP has clearly demonstrated quality of service, heterogeneity, low cost, complete development and user environments, support from the general industry and longevity. IP is over twenty years in the making and remains the primary worldwide interconnect. No other level 3 protocol can claim such diverse use and acceptance. IPv6 joins the level three protocol family with added performance and capability. SHARE would do well to employ IP as the interSAN communication protocol. The primary advantage the SHARE gains is a greater heterogeneous operation with a larger bodies of extant systems. As shown, IPv6 clearly addresses the issues identified in section 3.2 of non-SAN to SAN SHARE systems.

6.3 IPv6 high performance computing disadvantages

IPv6 is not specifically designed for use in embedded extreme performance, tightly coupled, massively parallel programming subsystems. That is not so say it can not be, or that it is not close in design, but that it was not the motivating factor for its inception. What limits IPv6 most are its extensive addressing fields. Eighty percent of the IPv6 header is dedicated to addressing. Such extensive addressing, deep in the header, is not necessary in intraSAN communication. It would be burdensome to continually skip those fields when parsing a packet. Consequently, IPv6 would not be the protocol choice for intraSAN communication as outlined in section 3.3. However, IPv6 may be fine interSAN or even SAN to conventional area network. IPv6, being a level three protocol, has bulk, but conveys enough service information content to justify the header size in relation to the kinds of data transmission expected. Yet, IPv6 still does not purport to be a high performance protocol, and hence, may not serve the needs of high performance embedded, massively parallel algorithmic designs.

The majority of SHARE communication occurs within the SAN and interSAN. Moreover, the majority of the SHARE algorithms are expected to be tightly coupled, massively parallel, high performance programs. Accordingly, such algorithms and programs would not use the services provided by IPv6 either intraSAN or interSAN.

Another issue is source routing. Communication expected intraSAN and interSAN heavily depends on source routing. Although IPV6 has such a capability, it is not instituted in such a fashion as to be useful by performance conscious programs. The IPv6 source routing is cumbersome. It requires the continued examining and splicing of the data packet. This means additional cost of parsing and coupling.

The result is IPv6 is too costly in terms of time and efficiency to be used wisely in the intraSAN and interSAN arenas and these are the two most performance critical dialogue regions of the SHARE.

6.4 IPv6 verses SHARE Criteria

The following table (Table 6-1) shows where IPv6 best fits the requirements of the SHARE communications model. It is apparent that the level three protocol does not meet any of the SHARE intraSAN communication needs. This limits IPv6 within the SHARE domain, potentially so much so that it may lack acceptance interSAN. However, IPv6 meets most of the SAN to conventional network communication considerations.

Table 6-1 SHARE IPv6 Protocol Requirements Cross Reference

SHARE	IPv6		
Protocol Criteria	Problem Set 1 *InterSAN*	Problem Set 2 *Conventional*	Problem Set 3 *IntraSAN*
Reliable	✓	✓	
Security	✓	✓	
Heterogeneity		✓	
High Performance			
Real Time			
Just-in-Time		✓	
Transparent Operation		✓	
Message Passing	✓	✓	
Cut-Through Routing	✓	✓	
Dynamic Discovery		✓	
Source Routing		✓	
Destination Routing	✓	✓	
Quality of Service		✓	
Robust	✓	✓	
Supportable	✓	✓	
Maintainable	✓	✓	
Extensible	✓	✓	
Voluminous Data Xfers	✓	✓	
Light Weight			
Scalable	✓	✓	

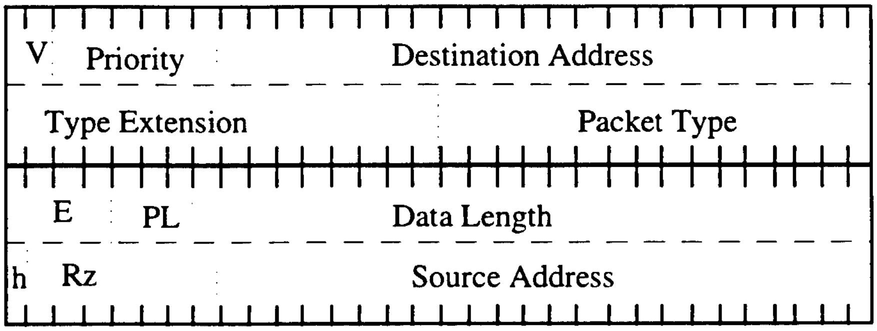
7. Packetway

Packetway is a working group under the auspices of the Internet Engineering Task Force. The idea for Packetway (formerly called Messageway) developed in response to perceived slow progress being made by the Internet Protocol Next Generation working group, especially in high performance networking. A splinter group within the IETF began developing a protocol that would compare to IPNG, but, would be designed for high performance, switched based networks, massively parallel computing and embedded systems applications. Advocates for Packetway present it as a level two protocol with integrated level three services.

7.1 Packetway Header

The Packetway header is currently being developed and proposed as an RFC. The format is shown in Figure 7-1. There are fields ranging from one bit to twenty-five bits, but the more important fields are placed at quadword boundaries. (A quadword boundary is one in which the physical address lowest two bits remain zero). In fact, the header is composed of two quadwords. The first quadword contains the most important service information followed by the higher level fields for those elements which need additional processing information. The first quadword is sufficient information for routing the packet to a specific node. This supports the Packetway claim to be a high performance level two protocol. However, most of today's hardware and network software drivers can not make optimal use of quadwords. Hence, as the data arrives, usually in thirty-two bit words, processing is performed in standard network big end first order.

Figure 7-1 Packetway Header Format



7.1.1 Version

The version is two bits and is the highest order field (lowest addressed bits) of the first quadword. Version is the first field to arrive. Given the version, decision processing may switch to specific handling functions that effectively decode the remaining information.

7.1.2 Priority

The priority is six bits and follows the version. The priority is bits fifty-six through sixty-one of the first quadword. This makes it the second field available for decoding. Given the priority, special handling may be applied to the following information. If the receiving element is an intermediate hop, priority switching may be enabled.

7.1.3 Destination Address

The destination address follows the priority and is bits thirty-three through fifty-seven. There are twenty-four bits of destination address. The destination address may be used to address a physical device as well as a logical device. Hence, an entire community of nodes may receive a message as a result of a logical address. However, like IPv6, Packetway is unable to address a single processing element within a node and is only capable of node addressing granularity. Should a node comprise a battery of processing elements or processes, there is no provision for addressing these individual SAN components.

7:1.4 Type-Extension

The type extension field constitutes bits thirty-one through sixteen. Though the type-extension field precedes the packet type field, it allows the receiving device foreknowledge when deciphering packet information. In addition, type-extension, in effect, displaces 'packet type,' the most important field within the first quadword, so that it is aligned on a quad word boundary.

7.1.5 Packet Type

The packet-type is the addressable base of the first quadword using little endian notation. The field is sixteen bits in length. Since the packet type field can be addressed directly as a short, it may be used as a switch to invoke packet-deciphering software or as a hash for a hardware accelerator. If additional processing granularity is required, the packet type and the type extension field immediately preceding it may be addressed as a single thirty-two-bit value.

7.1.6 Endianness

The four endianness bits are contained within the first field received of the second quadword of the Packetway header. If the receiving unit is capable of interpreting this field (employing hardware) spontaneous conversion may be performed on the payload of the packet. This may reduce latency of data processing especially in the event cross-endianness machines communicate frequently.

7.1.7 Packet Pad Length

Packetway references header fields in terms of eight byte words, which are referred to as quadwords. Packetway stipulates that all data transfers must be in quadword units. Hence, all applications that transfer data not quadword aligned must pad the remaining data to the nearest quadword boundary. Consequently, a field in the header is required to specify how much pad accompanies the data. The pad length field, therefore, is three bits to accommodate up to seven octets of pad.

7.1.8 Data Length

Packetway data length is recorded in terms of quadwords (eight bytes). A quadword therefore is the fundamental data abstraction in the Packetway packet. When payload data not meeting this requirement is passed, data padding is applied. The data length reflects the number of quadwords and the packet pad length field records the total number of pad octets needed to complete the last quadword.

7.1.9 Optional Header

The optional header field "H" in the Packetway header identifies the existence of additional headers following the Packetway header. These optional headers may be used for any purpose, but typically are used for Packetway routers which embed router-to router messages and annotations for special handling or processing. Some headers may also identify the existence of trailers that follow the payload. All optional headers, if present, are processed in the order in which they appear.

7.1.10 Reserved

There are seven reserved bits in the Packetway header, though unusual, it does allow for future enhancement.

7.1.11 Source Address

The source address is a twenty-four-bit field. It is the base address of the second quadword of the Packetway header. Its placement suggests it is not vitally important for peer to peer communication. However, for router to router communication, where source and destination addresses are stored, the source address is immediately addressable and quickly attained.

7.2 Packetway High Performance Computing Advantages

Packetway purports the ability to function faster for embedded and switch based networks. It claims to have low overhead, well ordered header fields and accommodations for hardware acceleration. Furthermore, it is presented as a level two protocol with built-in provisions for level three functional services. These features make Packetway appealing to embedded application designers who wish to minimize information transfer latency and increase information content.

Traditionally, the second layer of the network model provides merely point-to-point connectivity. Usually any further processing is begun at the network, or third layer. Packetway breaks with tradition and introduces service information and source routing below the network level. Hence, the power purported by Packetway is that service information does not impede or detract from the effectiveness of the layer two functions, while providing level three capabilities. The message being broadcast is that the Packetway is so well organized and well defined, that its merit is self evident.

7.2.1 Symbols

Any number of symbols may be placed in front of the Packetway header. As a Packetway packet moves through a network, these symbols, perhaps representing special handling instructions, can be interpreted at receiving network elements by special hardware or software. These symbols give Packetway the ability to affect the handling by network elements as the data is moved. Similarly, symbols enable the ability to use encryption as part of the protocol for systems that require such forms of security.

7.2.2 Hardware Acceleration

The Packetway header is designed to afford as much processing information as possible as early on as possible. This means the fields are sensitive to big endian network ordering. In other words, the big end of the frame arrives first. In the Packetway header, this means version first, followed by the priority and destination address, all contained within the first quadword. These fields are considered the most critical in decision processing of the packet.

Hardware accelerators may be programmed to read these fields, decipher a course of action and enhance the digesting of the frame even as it arrives from the network interface. Additionally, the second quadword may also be read and processed by hardware. Together, these two quadwords provide all the information any node interface needs to know. Of particular note is that the Packetway header is designed more for the intermediate hop than for the destination element. This is apparent by the order of importance given the fields.

7.2.3 Cut-Through Routing

The most important requirement of embedded switch-based systems is the need to move the data through the network as quickly as possible with minimal latency. To achieve this, cut-through routing was developed enabling the source of a packet to directly specify the route that the packet is to move. To support this function, hardware has been developed to accelerate the movement by using the leading frame information as a network vector. For those frames needing forwarding, the hardware would be induced to move the data out even as it comes in. Timely forwarding decisions, and timely forwarding, require early access to the destination address. To accommodate this, Packetway placed great importance on the location and size of the destination address. Being in the high order of the first quadword, and more importantly being the base address of the word, the destination address is efficiently addressable and quickly used.

7.2.4 Source Routing

Packetway touts true source routing. Source routing tends to be faster when the source and destination are known and information exchanges are frequent. The advantage of Packetway source routing is that the payload or header never need to be parsed to acquire the next hop. The next hop is always the next level two address transferred before the Packetway header. There may be any number of source routes present, each moving the data that much closer to its final target. The native network interface controllers strip the source route as the frame moves toward the destination. The result is the Packetway header, along with the payload is the only portion which arrives at the destination node.

7.2.5 Fast Intermediate Node Processing

A stumbling block to most level three protocols is the inability to handle hops well. There are many schools of thought as to how to handle intermediate processing. Packetway has resolved this difficulty by solving for the intermediate case. Packetway is formatted to provide an intermediate node (usually a router) as much information regarding the packet as possible, with out compromising data latency. The mechanism that enables such performance is built-in native network addressing. In other words, the ultimate advantage is that the destination receives a packet handled most expeditiously by the capabilities intrinsic to the native network. Moreover, when the packet arrives at an end node, it can be assured that, one, the information was meant to get the node, and two, the node got the information in the most expeditious manner possible.

What makes this possible is that Packetway is a level two and a half protocol. That means, it has all the function of the level two protocols and not all the overhead of a level three protocol. The benefit is added functionality. Once the data arrives, more is known about it at the lower layers; and, hence, as mentioned earlier, processing may be enhanced by use of parallel processing algorithms, hardware acceleration, etc.

The level three information conveyed in the header includes destination address, type-extension, packet type, endianness, pad length, data length, optional headers, and source address. These fields, when compressed to level two, enable intelligent network interface controllers the ability to parse, decipher and direct the frame quickly to its designated recipient.

7.2.6 Scalability

Packetway provides twenty-four bits of addressing, plus native routing and up to 248MB of payload. It is not likely networks will stress such figures in the near term.

7.3 Packetway High Performance Computing Disadvantages

Work has been slow in the adoption and specification of Packetway primarily because of the inability by the working group members to agree upon issues. The chair of the working group has been a strong influence, yet to date has no formal intention of implementing the specification. This among other issues has made the future of Packetway questionable.

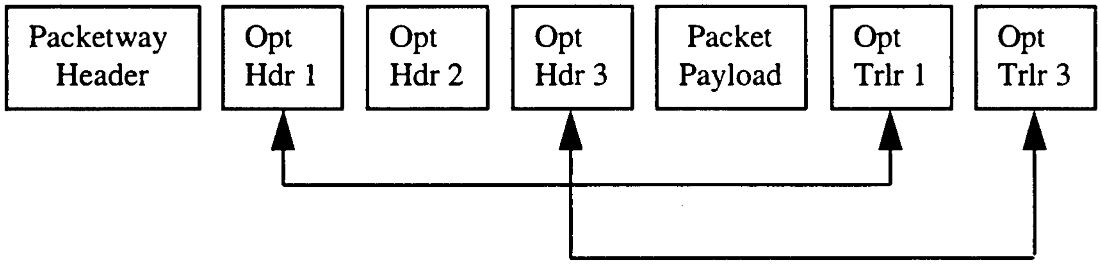
7.3.1 Packetway Features More than Routing

Although Packetway is purported to be a level two and a half protocol, it is clearly a level three protocol. When conveying anything more than connectivity (a data link function) additional processing is required. Packetway tries to hide the fact that it allows native routing prefixed to the header as an additional and unique capability. So do all other level three protocols. Packetway, however, goes beyond such widely accepted and acclaimed level three protocols as Internet Protocol and 803.3 by providing endianness, type extension and optional headers. Furthermore, Packetway uses quadwords where most networks today, and certainly embedded systems use thirty-two bit network connections. Only if traffic is moved along a sixty-four bit data bus, and then, only short distances, will the Packetway quadword header format be of any value. Then again, if the data are moved along a data bus, then the data are probably understood by the pairing processing agents and even a level two protocol would be too "heavy" a burden.

7.3.2 Optional Headers

Packetway includes optional headers and these optional headers may or may not have optional trailers. Given a Packetway frame, when an optional header, having a trailer, is followed by another optional header, having a trailer, then, as depicted below, the two trailers must appear in the same order as the optional headers Figure 7-2 Figure 7-2. Notice the awkward overlap. The reasoning is that should an optional header with a CRC trailer be followed by another optional header with a CRC trailer, the two CRC trailers appear in the order in which the hardware in

Figure 7-2 Optional Headers in a Packetway Frame



route would attach them. The headers may also be placed by hardware before the payload is transmitted, hence sandwiching the existing data frame.

The placement of the optional headers and trailers may produce undesirable side effects. For instance, the receiving node, may have to stack the headers with trailers, in order to process the pairs correctly. This would mean having to store information, which may not even be used by the node, just to ensure the data frame integrity as it passed through. This activity is certainly not something normally associated with a level two protocol. Moreover, by assuming or creating the burdens of optional headers and the processing required to handle them, a protocol cannot meet the light weight requirement of a high performance protocol.

7.3.3 Unknown Future

Although Packetway is still evolving, its adoption and subsequent acceptance is a matter of significant doubt. Packetway, another level 3 protocol with robust features, does not address any significant need, particularly within the Internet domain. There, any acceptance of Packetway is unlikely. At most, Packetway might eventually gain the support of a few universities and small stand-alone experimental networks. In addition, owing to its service information content, the employment of Packetway in high or extreme performance networks is doubtful. Similarly, Packetway has too weighty a header to be a viable alternative for switched based embedded systems. Uncertainty about where Packetway fits has been a cause for concern at the IETF. IETF includes Packetway detractors who don't understand Packetway and, so, oppose it. Of course, Packetway has supporters, among them Mississippi State University, but the interest there is contractual. Clearly, PacketWay's future is very uncertain.

7.4 Packetway verses SHARE Criteria

Figure 7-3 Packetway verses SHARE Criteria

SHARE	Packetway		
Protocol Criteria	Problem Set 1 InterSAN	Problem Set 2 Conventional	Problem Set 3 IntraSAN
Reliable	✓	✓	
Security	✓	✓	
Heterogeneity	✓		
High Performance			
Real Time			
Just-in-Time	✓	✓	
Transparent Operation			
Message Passing	✓	✓	
Cut-Through Routing	✓	✓	
Dynamic Discovery			
Source Routing	✓	✓	
Destination Routing	✓	✓	
Quality of Service			
Robust	✓	✓	
Supportable	✓	✓	
Maintainable	✓	✓	

SHARE	Packetway		
Protocol Criteria	Problem Set 1 InterSAN	Problem Set 2 Conventional	Problem Set 3 IntraSAN
Extensible	✓	✓	
Voluminous Data Xfers	✓	✓	
Light Weight			
Scalable	✓	✓	

8. Conclusion

The goal of this study has been to look at the SHARE protocol communication needs and requirements. Some time has been spent describing the transparent, lightweight, yet robust nature of the SHARE protocol criteria. Three domains were discussed, each with unique prerequisites to establish SHARE conformance. For this reason, two likely protocols have been presented, a ubiquitous protocol, IPv6, and, a developing protocol, Packetway.

The ultimate goal is to identify which of the two protocols is more amenable to all the SHARE problem domains. Clearly each has benefits expressed in terms of service information conveyed in the header. Similarly, how well that information translates into productive work for the application(s) is paramount. It is not as easy a case as looking at the headers and discounting one or the other based on size or configuration. The solution is more in terms of application productivity. That is, the ability for the application to perform its designated operations. For example, an application may have excessive input/output where another may be primarily computation. Likewise, if the application is presenting user data, the rate at which a user can assimilate the information may not be the rate at which algorithm is capable of generating the data. For this reason, the protocol used in one type of application may fall completely short in another. Perhaps multiple protocols may need to be employed. This is not necessarily all bad. For this very reason, routers have been used to do the translations. However, most commercial routers only perform protocol translation on known, and widely popular types of protocols. It is not likely that an emerging protocol, which in fact competes with an existing ubiquitous one, would even stand a chance of acceptance and use. As much as Packetway purports to be a level two and a half protocol it has been clearly demonstrated to be yet another level three protocol.

Where Packetway has the advantage over Ipv6 is that it provides more network services. This is also its disadvantage, because that translates to more header parsing and more decision making for end nodes interested in algorithmic function than information conveyance. So in the end, Packetway is actually less useful than IPv6. This is especially true at the intraSAN and to a leser extent the interSAN. Packetway completely breaks the SHARE ability to inter operate with existing systems that, one, either have no intention of adding a Packetway router service, or two, question a replacement for a current network protocol which is proven.

The dismissal of Packetway focuses attention on IPv6. Clearly Ipv6 has an advantage of popularity over Packetway, true, but little else. Again, IPv6 is an excellent protocol having existed for many years. It has more than proven its ability to endure, mutate and conform to uses never originally intended. However, IPv6 stands little chance in the embedded high performance, massively parallel application arena of the intraSAN. Like Packetway though, Ipv6 would fit better the interSAN communication domain. Here, the kind of information conveyed needs more service information, yet be adroit in its configuration of such. To that extent, IPv6 has it over Packetway because of software involvement needed to parse the Packetway header verses the IPv6 header. Similarly, Ipv6, being dominant and ubiquitous in the network industry today, would stand a better chance to piece SHARE applications into the marketplace.

In conclusion, neither protocol serves all of the SHARE problems. This is certainly true for the intraSAN, where high speed and few services information are needed. As for interSAN, where connectivity speed is still an issue, but services begin to emerge beneficially, Ipv6 or Packetway would suffice. However, here IPv6 has a slight advantage. IPv6 does not convey as many services (namely tags). Consequently, IPv6 would incur less software intervention, to parse, regulate and administrate. As for communication with conventional networks, Ipv6 clearly beats Packetway in acceptance and use. Here the services (endianess, type extension, tags, etc.) of Packetway may be of interest but certainly limited since a "high performance" is not the goal when communicating with a data base or graphic user interface. Bottom line, the protocol used on the SHARE program remains more subjective than quantifiable.

SHARE
CRYPTO PROTOCOL STUDY

Rev- , 12/2/97

L-3 Communications
One Federal Street
Camden, NJ 08102

SHARE CRYPTO PROTOCOL STUDY

TABLE OF CONTENTS

SHARE CRYPTO PROTOCOL STUDY

1.0 INTRODUCTION AND TASK DESCRIPTION:

Per Statement of Work, 8/7/97: "The SIM (SHARE Infosec Module) has been designed to operate using Secure Packetway as the communications protocol and using the Electronic Key Management Standards (EKMS) for key management and key distribution. This task will evaluate the SIM's ability to support a variety of other (emerging) standards, including ISAKMP, Space Communications Protocol Standards (SCPS), and IP Version 6. Study results will be documented in a study report."

The purpose of this study is to understand the issues associated with making the SIM compatible with the new standards. This report is a 'higher' level document and may be used as the starting point in the development of the detailed interfaces between the SIM and its associated Secure Router. However, this study provides more than just an exposure to the protocols, as the specific SIM requirements necessary for compatibility with each standard are delineated.

Much of this report is a discussion of the relevant sections of the referenced standards, and the discussions may be read for the most part without knowledge of SIM design. However, when the implications on SIM design are discussed, it is assumed that the reader is familiar with the basics of SIM design and operation.

The study results are partitioned into Sections 2, 3, 4, and 5 per each of the four standards considered, i.e. Packetway, IPv6, ISAKMP, and SCPS. The discussions repeat numerous passages from the referenced standards without quotes identifying such passages and their source. It is hoped that this unburdening of the text from continuous reference designations will allow this document to be more readable. Section 6 lists the three sections where the specific SIM requirements are located. Section 7 is a series of Appendices which may help in this document's understanding.

2.0 PACKETWAY / EKMS - SIM EDM DESIGN OBJECTIVES:

The SIM EDM (Engineering Development Model) is a PWA that plugs into a PC's PCI bus. The SIM is a component of a Secure Router which occupies other slots on the PCI bus. The SIM EDM is designed with the objective of demonstrating the major data performance attributes required of such a secure cryptographic module, including:

1. A compatible PCI interface with burst throughput consistent with current PCI bus capacity of 33 MHz @ 32 bits (i.e. 1.056 Gbps, 132 MBps), and average throughput consistent with PC implementation, i.e. via a single shared PCI bus.
2. Encryption / decryption via a Type 1 algorithm and device.
3. Encryption / decryption at a 40 MHz clock rate, i.e. a burst data rate of 160 MBps or 1.28 Gbps.
4. Session key agility.
5. Network (PCI bus) key load capability.
6. SHARE data format compatibility.
7. Support chip availability and compatibility, i.e. PCI Controller, custom FPGA, FIFO, RAM, PLD, clock source, etc., as required.
8. Compatibility with SHARE Key Management concepts.
9. Compatibility with traditional secure design concepts, e.g. access control, alarm generation, red/black separation, etc.

A comprehensive report on the key management aspects of a production deliverable SIM and of the SHARE-HPSC network in general has been prepared as "SHARE-HPSC Key Management Plan, Rev -1, 7 Nov, 1997". The report expands on additional features required of a Type 1 SIM.

3.0 IPv6 (Internet Protocol Version 6):

3.1 IPv6 Background:

IP provides network layer services to the Internet and the functions necessary for connecting networks and gateways together into a coherent system. It is responsible for delivering data from the source to the final destination - this it does with independence from specific routing and switching implementations. IP provides the functional and procedural means to set up and terminate a call, to route data, and to control data flow across the network. Other supporting protocols for route discovery and address mapping also reside with IP at this layer.

IP Version 6 (IPv6) is a new version of the Internet Protocol, designed as the successor to IP version 4 (IPv4). IPv6 is a work in progress with issues and implementation implications still under discussion. This report addresses the crypto module (i.e. the SIM), and the discussion herein draws from the current available IPv6 literature as referenced in Appendix 7.2. Although updates to the reference literature are anticipated, the considerable effort to date suggests that the references will not undergo wholesale change, and the conclusions drawn here with respect to SIM issues are expected to remain valid.

An informational note on the transition approach from IPv4 to IPv6: As with the existing IPv4 Internet backbone, the IPv6 backbone infrastructure will be composed of many Internet Service Providers (ISPs) and user networks linked together to provide the world-wide Internet. The "6bone" is a virtual network layered on top of portions of the physical IPv4-based Internet to support routing of IPv6 packets, as that function has not yet been integrated into many production routers. The network is composed of islands that can directly support IPv6 packets, linked by virtual point-to-point links called 'tunnels'. The tunnel endpoints are typically workstation-class machines having operating system support for IPv6.

The changes from IPv4 to IPv6 fall primarily into five categories:

1. Expanded addressing capabilities
2. Header format simplification
3. Improved support for Extensions and Options
4. Flow labeling capability
5. Authentication and privacy capabilities

By examining these categories it is seen that the SHARE red and black half routers are the functional elements that must insure IPv6 compatibility. However, Category 5, "Authentication and Privacy Capabilities", refers to services normally provided by the cryptographic module, and this category's effects on SIM compatibility are further examined.

The IPv6 specification designates its security requirements by a simple reference. Per IPv6: "This document specifies that:

1. -the IP Authentication Header [RFC-1826], and
2. -the IP Encapsulating Security Payload [RFC-1827], be used with IPv6, in conformance with
3. -the Security Architecture for the Internet Protocol [RFC-1825]."

The effect of these three "references" on the SIM design are discussed in succeeding sections.

It worthwhile to note the relationship and differences between Internet RFCs, Drafts, and Specifications. An RFC is a "Request for Comment" document normally containing a substantial amount of information about the subject, much of which is expected to be incorporated into the final specification, and which is issued for discussion and suggestions for improvement. From the "comments", the associated IETF Working Group develops a "Specification", the first versions of which are working documents called "Internet Drafts". As the specification proceeds from Internet RFC to Internet Specification, additions, modifications, and deletions are incorporated. In the following sections, both the RFC and the current Draft are discussed, as they both lend insight into the resulting requirements.

3.2 RFC 1826, "IP Authentication Header":

The IP Authentication Header (AH) provides data integrity and authentication. It may also provide non-repudiation if used with certain algorithms, e.g. by using an asymmetric digital signature algorithm such as RSA. The IP authentication algorithm is calculated over the entire IP datagram, and the AH may be applied to IPv4 and IPv6 datagrams. The AH is normally inserted after an IP Header and before the other information being authenticated. For the calculation, the authentication field itself and all fields that are modified in transit are treated as though they contain zeroes. Any and all other authentication fields are included in the authentication calculation. Data confidentiality must be provided by other means, such as the IP Encapsulated Security Payload (ESP).

In order for the AH to work properly without changing the entire Internet infrastructure, the authentication data is carried in its own payload. This payload may be used or ignored. Note that if an asymmetric authentication algorithm is used and the intermediate routers are aware of the appropriate public keys and the authentication algorithm, then these routers could authenticate the traffic without being able to forge or modify the message content. Note also that intermediate authentication requires path discovery since it is not possible to authenticate a packet fragment.

The implementation of an AH requires that Security Association (SA) parameters be negotiated and stored in a table. This table must be read by the crypto module to acquire the key, algorithm, and other parameters associated with the process of creating or verifying an AH. The diagram of Appendix 7.5 may help the reader understand the SA concept and related security parameter terminology.

The AH payload is a variable length field defined by parameters within the field itself. The fields within the AH are illustrated in the AH data format below:

```
      8 bits          8 bits           8 bits           8 bits
+------------------+------------------+-----------------  +-----------------+
| Next Header  |    Length     |                 Reserved             |   32 bits
+------------------+------------------+------------------+-----------------+
|                   Security Parameters Index (SPI)                   |   32 bits
+------------------+------------------+------------------+-----------------+
|                                                                      |   32 bits
+      Authentication Data (variable number of 32-bit words)         +
|                                                                      |   32 bits
+------------------+------------------+------------------+-----------------+
```

Figure: Authentication Header Syntax

-Next Header - identifies the next payload
-Length - the length of the Authentication Data field in 32-bit words.
-Reserved - reserved for future use
-Security Parameters Index (SPI) - PR value identifying the security association for this
datagram (values 0 - 255 reserved). Note that an implementation will normally use a combination of the Destination Address and SPI to locate the Security Association which specifies the field's size and use.
-Authentication Data - an integral number of 32 bit fields defined by "Length".
Padding to the "Length" value is required. Max length = [(2^8)-1)]*32 = 8160 bits = 1020 bytes.

Implementations that claim conformance to this RFC 1826,

1. MUST fully implement the described header,
2. MUST support manual key distribution for use with this option,
3. MUST comply with all requirements of the "Security Architecture for the Internet Protocol (RFC 1825, August 1995), and
4. MUST support the use of keyed MD5 with up to 128 bit key as described in "IP Authentication using Keyed MD5, RFC 1828, August 1995." (Longer secret authentication key lengths are encouraged. Implementations MAY also implement other authentication algorithms.).

3.2.1 Internet Draft, "IP Authentication Header":

The current IP AH Internet Draft contains requirement differences from RFC 1826 that are implementation mandatory:

1. MUST fully implement the AH syntax and processing described in this Internet draft.
2. MUST comply with all requirements of the Security Architecture document,
3. MUST support the following algorithms:
   -HMAC with MD5 (see "The Use of HMAC-MD5-96 within ESP and AH", Internet Draft, 7/2/97, Madson & Glenn).
   -HMAC with SHA-1 (see "The Use of HMAC-SHA-1-96 within ESP and AH", Internet Draft, 7/2/97, Madson & Glen).
4. Also, if the key used to compute an Integrity Check Value (ICV) is manually distributed, correct provision of the anti-replay service would require correct maintenance of the counter state at the transmitter, until key is replaced, and there would likely be no automated recovery provision if counter overflow were imminent. Thus, a compliant implementation SHOULD NOT provide this service in conjunction with SAs that are manually keyed.
   Note: the "counter state" refers to the value in the Sequence Number Field of the AH format. The counter value monotonically increases and is always present even though the receiver may elect not to act upon it and thus not enable the anti-replay service for a specific SA. The sender's and receiver's counters must never be allowed to "roll over". A new SA and new key must be established prior to the transmission of the 2^32 packet of a particular SA.

These changes do not modify the character of RFC-1826.

3.3 RFC 1827, "IP Encapsulating Security Payload (ESP)":

ESP is a mechanism for providing data integrity and confidentiality and may be used to encrypt either a transport-layer segment, e.g. TCP, UDP, ICMP, IGMP, or an entire IP datagram. Encapsulation of the protected data is necessary to provide confidentiality for the entire original datagram. ESP may also provide authentication to IP datagrams in some circumstances, depending on which algorithm and algorithm mode are used.

Key management is not included in the ESP specification. The only coupling between the key management protocol and the security protocol is via the SPI and its contents, i.e. the SA. A separate key management mechanism is required to negotiate a number of parameters for each SA, including not only the keys but other information, e.g. the cryptographic algorithms and modes, security classification level, etc. An ESP implementation will read the security parameter table created by the key management protocol to determine how to process each datagram, e.g. which algorithm, mode, key, etc. to use.

A high level diagram of a secure IP datagram follows:

```
|<--------------Unencrypted------------->|<-----------------Encrypted------------------->|
+-------------+----------------------+----------------+---------------------------------------+
| IP Header | Other IP Headers | ESP Header |            encrypted data            |
+-------------+----------------------+----------------+---------------------------------------+
                                       |                |
 ____________________________/                \________________________
/                                                                      \
|                    The ESP Header is expanded as follows:              |


|<---------Unencrypted---------->|<-----------------Encrypted------------------------>|
+-----------------------------------+-------------------------------------------------------+
|         SPI (32 bits)          | Transform (algorithm) data (variable length)  |
+-----------------------------------+-------------------------------------------------------+
```

Implementations which claim conformance to this RFC-1827:

1. MUST fully implement the described header,
2. MUST support manual key distribution with this header,
3. MUST comply with all requirements of the "Security Architecture for the Internet Protocol, RFC-1825, Aug, 1995".
4. MUST support the use of DES in CBC mode. (as specified in "The ESP DES-CBC Transform, RFC-1829, 8/1995".
5. Implementations of ESP claiming to support multi-level security systems MUST support implicit labels, and MAY support explicit labels. Labels are viewed as implicit in IPv6 because the SPI implicitly designates the sensitivity level. IPv4 normally requires the explicit designation of sensitivity level in those systems which require security labels. IPv6 users may still choose to carry explicit labels, although a given implementation is not required to do so.

Other ESP transforms (algorithms and modes) may be used.

Implementations SHOULD support user-oriented keying as described in the IP Security Architecture document. If user-oriented keying is not in use, then the algorithm in use should not be an algorithm vulnerable to any kind of Chosen Plaintext attack. Chosen Plaintext attacks on DES have been described in the literature. Use of user-oriented keying is recommended in order to preclude any sort of Chosen Plaintext attack and to generally make cryptanalysis more difficult.

3.3.1 Internet Draft, "IP Encapsulating Security Payload (ESP)":

The current IP ESP Internet Draft contains requirement differences from RFC-1827 that are implementation mandatory:

1. MUST comply with the ESP syntax and processing described in this Internet draft,
2. MUST comply with all the requirements of the Security Architecture document,
3. MUST support the following algorithms:
   -DES in CBC mode (see "The ESP DES-CBC Cipher Algorithm With Explicit IV", Internet Draft, 7/2/97, Madson & Doraswamy
   -HMAC with MD5 (see "The Use of HMAC-MD5-96 within ESP and AH", Internet Draft, 7/2/97, Madson & Glenn).
   -HMAC with SHA-1 (see "The Use of HMAC-SHA-1-96 within ESP and AH", Internet Draft, 7/2/97, Madson & Glen).

These requirement changes do not modify the character of RFC-1827.

3.4 RFC 1825, "Security Architecture for the Internet Protocol":

The architecture described in this RFC focuses on IPv4 and IPv6 mechanisms for IP-layer security, and is not intended to be an all encompassing Security Architecture for the Internet document. For instance, key management is not included. However, the IP layer is the Internet network layer and parallels the SHARE network protocol, Packetway.

It is envisioned that several key management systems will be used to support IPv6, including manual key distribution. "Work is ongoing within the IETF to specify an Internet Standard key management protocol."

3.5 SIM Requirements Summary for IPv6 Compatibility:

IPv6 includes protocols which can provide integrity, confidentiality, authentication, and non-repudiation for an IP datagram. Data formats are specified in support of these protocols. The mechanisms for key management and distribution are not specified, except to say that manual key distribution must be supported.

The SIM is in effect a hardware accelerator supplying cryptographic services for the SHARE secure router. The router provides (1) public network (IPv6) and (2) local SAN data format compatibility. In order for the SIM to be IPv6 compatible, it must supply the services that the router may call. As described above, the SIM services are specified by the supplied SPI. The SPI is a PR number which points to the SA to be invoked for the call. To be IPv6 compatible, the SAs stored in the SIM must support the functionality listed below. Recall that all formatting required by IPv6 is provided by the router function:

1. -Must support AH and ESP headers with the prescribed syntax.
2. -For the AH, must support the use of HMAC with MD5 and SHA-1.
3. -For the ESP, must support the use of DES in CBC mode, and HMAC with MD5 and SHA-1.
4. -For the AH, an ICV must be properly implemented.
5. -Multi-level security systems MUST support implicit labels.
6. -Must support manual key distribution.
7. -Must comply with all requirements of the IP Security Protocol:
    1. -Must support manual configuration of SAs.
    2. -Given two endpoints, it must be possible to have more than one concurrent SA for communications between them.
    3. -All such implementations must permit the configuration of host-oriented keying, i.e. one key per host, with possible multiple users per host. User oriented keying is preferred.
    4. -Must take reasonable steps to protect the keys and other SA information from unauthorized examination or modification.

These functional requirements are expanded below:

1. AH & ESP Headers:

IPv6 formatting compatibility is to be provided by the SHARE router. The SIM design supports the SHARE / SIM Data Format illustrated in Appendix 7.3.

2. HMAC Requirements:

The SIM EDM design does not include an authentication algorithm, although the SIM data format of Appendix 7.3 includes provision for header and data authentication codes. MD5 and SHA-1 need to be incorporated.

3. DES/CBC:

The SIM EDM contains a Type 1 classified encryption/decryption algorithm. To be IPv6 compatible, DES/CBC would have to be added, as would MD5 and SHA-1 authentication.

4. AH and ICV:

A properly implemented ICV must take into account the distribution method by which the AH key was received.

5. MLS & Implicit Labels:

An SA is intended to include the security information relating to a given network connection or set of connections. The SIM EDM does not incorporate the many security parameters normally included in a full SA implementation, e.g. authentication algorithm and mode and keys, encryption algorithm and mode and keys, initialization vector presence and size, lifetime of keys and SAs, etc. A complete SA implementation will enable the SIM to select from a variety of functionality as called by the router. Currently, only session keys have been incorporated in the SA to allow demonstration of SIM key agility.

6. Key distribution:

The SIM incorporates a rudimentary key load approach. Keys are loaded into the SIM from main memory via CPU controlled transfer. A transfer obeys the SHARE/SIM key transfer format. Also, a comprehensive "SHARE Key Management Plan" defines the methods of key distribution foreseen for the SIM and for the SHARE network.

7. IP Security Architecture Protocol:

The Security Architecture document contains justification and informational content for the above items 1 - 6, and includes additional tutorial discussion that might be applied to an IPv6 security design, i.e. automated key distribution, use with firewalls, etc. The document also contains the following specific requirements:

7-1. Manual configuration of SAs: As mentioned in 5 above, the SIM EDM does this in the most limited sense, i.e. the session keys are the only security parameter currently identified with an SA. The session keys are loaded into the SIM from main memory via CPU control.

7-2. More than one possible concurrent SA for communication between two end points: This is currently possible with the SIM EDM. It is only necessary for the router to call the proper memory address where the key (and subsequently, the SA) reside.

7-3. Host oriented keying: The designation of key orientation, e.g. host, user, is determined by the router function and is transparent to the SIM. A particular key loaded into the SIM memory may be host, user, group, etc., associated.

7-4. Key and SA protection: No key or SA protection is currently provided by the SIM EDM. A production SIM will have to support key and SA protection as defined by the security policy.

The SHARE router is the functional element that provides data format compatibility with IPv6 and other possible public network interfaces. A simplified data format has been implemented between the SIM and the router which shields the SIM from most of the complexities of the public network formats. The SIM/router format is outlined in Appendix 7.3. The main simplifying features of this format are embodied in the MWC, MES, HMS, and DMS fields. Very simply stated, the field values fix the points at which the SIM begins and ends the encryption/decryption processes. The Mode Code and SCID Address are also used directly by the SIM and further simplify SIM operation. The SIM is then in effect a hardware accelerator supplying cryptographic services for the SHARE secure router. Note: SPI, SAID (Security Asociation Identifier), and SCID (Security Context Identifier) are equivalent designations used by IPv6, SCPS, and Packetway respectively.

Note that the Data section of the format contains both encrypted and unencrypted partitions. Header and Data MACs are also embedded in the Data block to accomplish authentication objectives. Authentication failure at the receiver causes the block to be immediately discarded. A SIM production design implementation of the SIM will record relevant authentication failure information in the audit function.

Although not MANDATORY, the router and SIM implementations MAY and SHOULD support other functionality as described herein and in the relevant IPv6 specifications.

The key management data used by the IP-layer is presumed to have been managed by a higher level protocol, e.g. UDP, TCP. Work is ongoing within the IETF to specify an Internet Standard key management protocol.

4.0 ISAKMP (Internet Security Association and Key Management Protocol):

4.1 ISAKMP Requirements:

ISAKMP proposes a method for SA and key management to support IPSEC and IPv6. It provides (1) specific protocol support for the negotiation of security attributes, and (2) a framework for Internet key management. It has been recognized that authentication and key exchange must be combined for better security to include SA exchanges. ISAKMP SA exchanges provide network peers with the security functionality that enables an authenticated and protected agreement to a common set of security attributes. ISAKMP is distinct from key exchange protocols in order to cleanly separate the details of SA and key management from the details of key exchange. To this end, ISAKMP defines the procedures for:

(1) -creation and management of SAs,
(2) -authenticating a communications peer,
(3) -key generation techniques, and
(4) -threat mitigation (e.g. denial of service and replay attacks).

The SA protocol negotiates, establishes, modifies, and deletes SAs and their attributes. SAs contain all of the information required for the execution of various network security services. The key management protocol handles public key generation for the Internet community at large and private key requirements for those private networks with that requirement. ISAKMP does not establish session keys, however it can be used with various session key protocols, such as Oakley, to provide a complete solution to Internet key management.

4.2 ISAKMP and the SIM EDM:
The SIM EDM does not directly incorporate key (or SA) generation or a key (or SA) exchange function. In the SIM EDM, session keys are loaded into the SIM via the PCI bus under router CPU control and per the SHARE / SIM data format for key transfer. These keys are called by the router at the time of an outgoing/incoming call. This is accomplished by the router's specifying of the SIM's memory address where the key is located. In production versions of the SIM, this memory address will be the starting address of the full SA to be applied to the call. The remainder of this section discusses the features of ISAKMP which are planned for the SIM production version.

4.3 ISAKMP and Production SIM:
ISAKMP requirements are spread throughout the ISAKMP document. The interpretation followed here is that the ISAKMP document uses lowercase 'must' and other similar lowercase requirement statements as 'tutorial verbage', and uppercase statements such as 'MUST' as 'requirements verbage'. However, the ISAKMP document apparently is not rigorous with respect to defining uppercase 'MUST' and related requirements. This is particularly the case in those sub paragraphs defining the payload formats - all items of which "MUST" be adhered to, but which are not stated as such. This lack of rigor is alluded to in ISAKMP paragraph 1.1, Requirements Terminology, by "........ the words that are used to define the significance of each particular requirement are usually capitalized." (author's underline). In order to deal with this requirements uncertainty and to facilitate the understanding of the ISAKMP document, all ISAKMP uppercase requirements have been extracted and listed in Appendix Section 7.4. These requirements were examined and are summarized in the following sections.

As with the case of IPv6, most of the requirements are formatting requirements and fall within the domain of the red and black half routers.

4.4 SIM Requirements Summary for ISAKMP Compatibility:
In general - to be compatible with ISAKMP, the SIM must add the following capabilities:

1. Incorporation of multiple SAs for each of the security protocols offered, e.g. IP-ESP, IP-AH, etc.
    1. The SAs are to be loaded and stored in SIM memory, and are recalled by the router via the SPI.
    2. The SA information so stored must contain the algorithms and keys employed, and should also contain key and SA lifetimes, the SA source address(es), and sensitivity level(s).
    3. Note: The equivalent Packetway terminology for SPI and SA are SCID (Security Context Identifier) and SC (Security Context).
2. Recogition of SPIs and calling the appropriate SAs,
3. Provision for the possibility that an invalid SA may be called,
4. The allocation and partitioning of memory for SAs, and the mapping of SAs via SPIs,
5. Format compatibility with the router functions and its interface.

In general - to be compatible with ISAKMP, the router must add the following elements:

1. ISAKMP designated message and data formatting,
2. SA and SPI concepts,

3. Key Management protocol,
4. the described payloads and payload processing and
5. the described message exchange types (for exchanging key generation and authentication data).
5. the Protection Suite concept,
6 the DOI concepts,
7. the required IANA designations.

5.0 SCPS (Space Communications Protocol Standards):

5.1 SCPS Approach:

The goal of the SCPS is to provide a suite of standard data handling protocols which (from a user viewpoint) make a remote space vehicle appear to be just another "node on the Internet. The easiest, lowest risk, and most direct way to extend Internet connectivity into space was to adapt the protocols that are used on the ground. SCPS adopted a policy of using COTS-supported standards wherever possible and to thereby capitalize on established user interface familiarity and minimize software development costs. This approach minimizes risk by exploiting the extensive operational experience that the Internet protocols have accrued.

Extensions to the Internet protocols were required because of physical and operational differences, including:

-Physical:
- -space link delays ranging from milliseconds to hours
- -potentially noisy space data links
- -limited space link bandwidth
- -sub-network types ranging from simple busses to LANs and WANs
- -interruptions in the end-to-end data path due to the uniqueness of the space environment (e.g. SEUs, intense bursts of noise) and of the effects of space unique hardware (e.g. spacecraft antenna obscurations).

-Operational:
- -the inherently sporadic nature of space-ground contact
- -tele-operations and maximum latency accomodatable

-On-board Resource Limitations:
- -limited processing power
- -limited program memory
- -limited data buffering
- -extreme asymmetry in bandwidth between forward and return links

5.2 SCPS Specifications:

Except for a very narrow range of conditions, the Internet off-the-shelf protocols do not satisfy requirements which address the space mission environment. A key element designed into the SCPS data formats is bit efficiency, i.e. minimization of overhead (less overhead is used than in the parallel Internet protocols).

The SCPS protocols include:
- SCPS File Protocol, SCPS-FP (based on Internet ***FTP***)
- SCPS Transport Protocol, SCPS-TP (based on Internet ***TCP***)
- SCPS Network Protocol, SCPS-NP, (based on Internet ***IP***)
- SCPS Security Protocol, SCPS-SP, (based on Internet ***IPSEC-ESP, IPSEC-AH***, and SDNS "SP3", ISO NLSP, I-NLSP)

Only the security protocol, SCPS-SP, will be further examined here as that is the protocol which affects the SIM.

5.3 SCPS Security Protocol, SCPS-SP:

5.3.1 General Description:

The Security Protocol provides security services on an end-to-end basis, i.e. from the source of the transmitted data to its destination. It is physically located between the network and transport layers, i.e. between layers 3 and 4 of the OSI Basic Reference Model. SCPS-SP provides confidentiality (encryption), integrity, and authentication services. Intermediate systems, (e.g. routers, gateways, etc) do not have access to the protected data unless explicitly authorized, and data routed and relayed (at the network layer) may not be viewed or altered. Access control is provided as a by-product of confidentiality and authentication as facilitated via the SA database.

Because address information is not enciphered (data format discussed later), SCPS-SP does not provide protection against traffic analysis, nor jamming or low-probability-of-intercept (LPI). Traffic analysis protection must be provided at the link layer, and jamming and LPI protection must be provided at the physical layer. SCPS-SP also does not provide protection against replay attacks. It is assumed that either the encryption algorithm or a sequence number provided by the upper-layer transport protocol (e.g. SCPS-TP) would protect against replay attacks.

Note that confidentiality, integrity, and authentication may also be provided at the physical layer. However, for store and forward systems, the security services at this layer can only be implemented on a hop-by-hop basis. This means that when using link layer security services, the data must be deciphered, exposing the data, and then re-enciphered at each hop. This is not the case when using the end-to-end security services provided by SCPS-SP.

5.3.2 IP Similarities:

SCPS-SP does not dictate the use of any particular cryptography, algorithms, or key management. Each system is free to adopt a security policy that provides an appropriate degree of protection for its data. SCPS-SP does not provide protection against traffic analysis or the interception of enciphered data. Where such prevention is required, the system must use link or physical protection services.

5.3.3 SCPS-SP Functional Requirements:

A simple statement of the SCPS-SP functional requirements for the protection of data between end points within a space data communications system is given below:

1. Access Control: Only authorized users (or processes acting on behalf of users) shall be granted access to network resources (e.g. end systems, transport protocols within an end system, and router).
2. Source authentication: the capability to verify the identify of the end system that originated the communication.
3. Command authentication: the ability to digitally sign a message to indicate that it was actually sent by the user claiming to send it.
4. Integrity: Ensure that the data sent is exactly the data received. Unauthorized modifcations shall be detected.
5. Confidentiality: Ensure that the data can be interpreted only by authorized users.

5.3.4 SCPS-SP Data Format:

```
                    |<--------------------------------Encrypted----------------------------------->|
                    |<-------------------------- ICV calculation ---------------------->  |
+. . . . . . +------------+------------+------------+-------------------------------------------+-----------+
: Network  | Security  | Security   | Transport |                                           | Integrity |
: Layer    | Protocol  | Protocol   | Layer     |              User                         | Check     |
: Header   | Clear   _ | Protected  | Header    |              Data                         | Value     |
:          | Header    | Header     |           |                                           | (ICV)     |
+. . . . . . .+------------+------------+------------+-------------------------------------------+-----------+
   8 bits    8 bits (min)                                                                      variable
               + options
```

Figure: SP-PDU

The SCPS-SP data format above illustrates how the functional requirements are satisfied. SCPS-SP encapsulates the transport layer data (T-PDU) into a security layer PDU (SP-PDU). The Clear Header provides routing information to the security protocol. The protected header contains information which may be enciphered along with the user data (e.g. upper layer protocol headers plus user payload data) depending upon the security policy being enforced by the SCPS-SP as well as the user's security services request. The ICV is calculated over the indicated fields and is itself encrypted. The lower level network layer then adds its header as shown.

5.3.5 SCPS-SP Processing:

When a packet is to be transmitted, the SCPS-SP:

- -receives a PDU from an upper layer protocol (e.g. SCPS-TP),
- -attempts to identify an SA based on the source and destination address,
  - -Notes: -the SAs are contained in a "database"
    - -if no SA entry is found,
      - -an SA or key management protocol (~ISAKMP) must first establish an SA, or
      - -if manual pre-placement of attributes is used, the SA entry <u>must</u> exist.
- -applies requested (or required) security services (e.g. confidentiality, integrity, authentication),
- -sends the PDU to the next lower protocol (e.g. SCPS-NP) for transmission over the network.

When a packet is received, the SCPS-SP:

- -receives a PDU from a lower level protocol (e.g. SCPS-NP),
- -identifies SA database entry, or discards the PDU,
- -based on security attributes of the SA,
  - -deciphers the PDU, and/or
  - -checks the PDU integrity, and/or
  - -authenticates the explicit source address, and/or
  - -checks the classification of an explicit label against that allowed on the connection by the SA.
  - -passes the PDU to the next upper layer protocol (e.g. SCPS-TP).

A most useful section in the SCPS-SP is Annex C. Annex C contains a checklist (compliance matrix) of the mandatory and optional requirements of implementations that claim to support the SCPS Security Protocol. For each requirement, the requirement source is indicated. The nine (9) page checklist is not repeated here, but its use in the allocation of SCPS-SP requirements to a SIM h/w or s/w element must be part of a SIM/SCPS design process.

5.4 SIM Requirements Summary for SCPS-SP Compatibility

In general, the SCPS-SP requires of the SIM the same elements that IPv6 requires of the SIM. The SCPS-SP protocol provides integrity, confidentiality, and authentication for Transport PDUs. A specific data format is specified in support of the SCPS-SP protocol. The actual encipherment algorithms and mechanisms for SA and key management and distribution are not specified. The choice of a specific security policy, and therefore the protection that will be achieved by the SCPS-SP user, is a local matter for determination by the security domain administration.

6.0 SPECIFIC SIM REQUIREMENTS (SECTION LOCATIONS):

- IPv6 compatibility requirements are delineated in Section 3.5.
- ISAKMP compatibility requirements are delineated in Section 4.4.
- SCPS compatibility requirements are delineated in Section 5.4.

7.0 APPENDICES

7.1 Acronyms:

3DES	-Triple DES
6bone	-IPv6 backbone (a virtual network layered on top of portions of IPv4)
ACT	-Anti-Clogging Token ("cookie")
AH	-(IP) Authentication Header
CA	-Certificate Authority
CBC	-Cipher-Block Chaining
CCSDS	-Consultive Committee for Space Data Systems
D-H	-Diffie-Hellman (key generation algorithm, via public cryptography)
DA	-Destination Address
DES	-Data Encryption Standard
DNS	-Domain Name System
DNSSEC	-DNS Security Extensions
DOI	-Domain of Interpretation
DSS	-Digital Signature Standard
EDM	-Engineering Development Model
ESP	-(IP) Encapsulating Security Payload
HMAC	-Hashed MAC (H=Hash, 'generic'). Must also include a specific MAC, e.g. HMAC-SHA-1.
I-NLSP	-Integrated Network Layer Security Protocol
IANA	-Internet Assigned Numbers Authority
ICMP	-Internet Control Message Protocol
ICV	-Integrity Check Value
IETF	-Internet Engineering Task Force
IMP	-Interface Message Processor
IP	-Internet Protocol
IPng	-IP next generation = IPv6
IPoS	-Internet protocols Over Satellite
IPRA	-Internet Policy Registration Authority
IPSEC	-Internet Protocol Security (IETF working group)
IPSO	-IP Security Option
IPv6	-Internet Protocol version 6
ISAKMP	-Internet Security Association Key Management Protocol
ISO	-International Organization for Standardization = International Standards Organization
ISP	-Internet Service Provider
IV	-Initialization Vector
KDC	-Key Distribution Center
LPI	-Low Probability Intercept
MAC	-Message Authentication Code
MD5	-Message Digest 5
MISSI	-Multilevel Information System Security Initiative
MLS	-Multilevel Security
MTU	-Maximal Transmission Unit
NLSP	-(ISO) Network Layer Security Protocol
OSI	-Open Systems Interconnection
PCA	-Policy Certification Authorities
PCI	-Peripheral Component Interconnect
PDU	-Protocol Data Unit
PGP	-Pretty Good Privacy
PKIX	-Public Key Infrastructure
PR	-Pseudo-random

PWA	-Printed Wiring Assembly
RFC	-Request for Comment
RSA	-Rivest-Shamir-Adleman (can denote an encryption or digital signature algorithm)
SA	-Security Association
SCPS	-Space Communications Protocol Standards
SDNS	-Secure Data Network System
SDNS-SP3	-SDNS Secure Protocol 3
SEU	-Single Event Upset
SHA-1	-Secure Hash Algorithm 1
SIM	-SHARE Infosec Module
SPI	-Security Parameter Index
TTP	-Trusted Third Party
UDP	-User Datagram Protocol (use port 500 for ISAKMP send/receive)
UNINETT	-
VPN	-Virtual Private Network
X.500	-a directory standard (supports email directory consolidations)
X.509	-part of X.500 that deals with authentication frameworks for directories (certifcates)

7.2 References Bases:

IPv6:

http://playground.sun.com/ipng/html
http://www.ietf.org/html.charters/ipsec-charter.html
http://www.ietf.org/ids.by.wg/ipsec.html

ISAKMP:

http://www.ietf.org/ids.by.wg/ipsec.html

SCPS:

http://www-scps.jpl.nasa.gov/scps/html/documents.html

PACKETWAY:

http://www.ietf.org/html.charters/pktway-charter.html

7.3 SHARE / SIM DATA FORMAT:

The SIM is a component of a SHARE secure router. The network layer protocol is implemented in software in the equipment's red and black routing functions. The router provide to the SIM the simple xmit/receive format below. This allows the SIM to be independent of the equipment's implemented network protocol, e.g. IP, Packetway, etc. and places little additional formatting demand on the router. Two additional formats are associated with data bypass and key data transfer.

A high level diagram of a secure SIM datagram follows:

```
|<--------------------------------------U----------------------------------------->|<---------E U E U--------->|

+--------------+-------+-------+---------+---------+----------+---------+---------+--------------------+------------------------------+
|    Synch     |Mode   |SCID   | SCID    | MWC     | MES      | HMS     | DMS     |         SV         |             Data             |
|              | code  | code  | addr    |         |          |         |         |                    |                              |
+--------------+-------+-------+---------+---------+----------+---------+---------+--------------------+------------------------------+
|     32b      |  8b   |  8b   |  16b    |  16b    |  16b     |  16b    |  16b    |        var         |             var              |
                                                                                       =<2^8By              Msg=<(2^16)*64b
```

Where U=unencrypted, E=encrypted.
Note that the Data section contains both encrypted and unencrypted partitions. Header and Data MACs are also embedded in the Data block to accomplish authentication objectives. Authentication failure causes the block to be immediately discarded. A final design implementation of the SIM will record relevant authentication failure information in the audit function.

Synch: fixed word synch preamble
Mode code: message mode identifier, e.g. xmit, receive, bypass, key
SCID code: Security Context Identifier. Same as IP SPI.
SCID addr: The address in SIM memory where the SCID data is located.
MWC: Message Word Count - message length in 64 bit words, beginning with the 4th word (04).
MES: Message Encryption Start - the point in the message where encryption starts, modulo 64 bits
HMS: Header MAC Start - the point in the message where the header MAC starts, modulo 64 bits.
DMS: Data MAC Start - the point in the message where the Data MAC starts, modulo 64 bits.
SV: State Variable - a PR number which supports the cryptographic process.
Data: This 'block' contains a variety of data, i.e. routing info, header and data MACs, data to be encrypted (original datagram, trailer, etc).

The main simplifying features of this format (as far as the SIM is concerned) are embodied in the MWC, MES, HMS, and DMS values. These values fix the points at which the SIM begins and ends encryption/decryption. The Mode Code and SCID Address are also used directly by the SIM and further simplify SIM operation. The SIM is then in effect a hardware accelerator supplying cryptographic services for the SHARE secure router.

7.4 ISAKMP 'Uppercase Requirements' Summary:

To begin the understanding of the ISAKMP document, all ISAKMP uppercase requirements are summarized below. The paragraph numbers are those of the ISAKMP document where the requirement appears. Paragraph headings have been retained even though no requirement is contained in them in order to give the reader the proper perspective of the requirement. Where the requirement text has not been included here, the number of uppercase requirements contained within the topic is indicated within parentheses, i.e. as (*). Note that an uppercase requirement could be a MUST, SHOULD, MAY, RECOMMEND, etc.

A parsing of the full ISAKMP document has been done with respect to uppercase requirements. These requirements have been extracted and listed below.

-1. Introduction
 -1.4 Security Associations and Management
  1.4.2 ISAKMP Requirements
   *1. SA establishment MUST be part of the key management protocol defined for IP based networks.
   *2. The basic set of SA attributes that MUST be implemented to provide ISAKMP interoperability are defined in Appendix A (of the ISAKMP document).
   Author Note: The ISAKMP Appendix does not include the basic set of SA attributes.
 -1.5 Authentication
  1.5.3 ISAKMP Requirements
   *1. Strong authentication MUST be provided on ISAKMP exchanges.
   *2. A digital signature algorithm MUST be used within ISAKMP's authentication component.
 -1.6 Public Key Cryptography
  1.6.2 ISAKMP Requirements
   *1. An authenticated key exchange MUST be supported by ISAKMP.
   *2. Users SHOULD choose additional key establishment algorithms based on their requirements.

-2. Terminology and Concepts
 -2.1 ISAKMP Terminology
  *1. DOI: A system SHOULD support multiple DOIs simultaneously.
 -2.4 Identifying Security Associations
  *1. An 'x' in the column means that the value MUST be present. (See para 2.4 in ISAKMP document).
  *2. This is only applicable for a phase 2 exchange and the value SHOULD be 0 for a phase 1 exchange because the combined cookies identify the ISAKMP SA.
  *3. During SA establishment, a SPI MUST be generated.
 -2.5 Miscellaneous
  2.5.1 Transport Protocol
   *1. Implementations MUST include send and receive capability for ISAKMP using the UDP on port 500.
   *2. Implementations MAY additionally support ISAKMP over other transport protocols or over IP itself.
  2.5.2 RESERVED Fields
   *1. All RESERVED fields in the ISAKMP protocol MUST be set to zero (0) when a packet is issued.
   *2. The receiver SHOULD check the RESERVED fields for a zero (0) value and discard the packet if other values are found.
  2.5.3 Anti-Clogging Token ("Cookie") Creation
   *1. The details of cookie generation are implementation, but MUST satisfy these basic requirements (originally stated by Phil Karn).

-3. ISAKMP Payloads
 *1. Additionally, all ISAKMP messages MUST be aligned at 4-octet multiples.
  3.1 ISAKMP Header Format:

-Major Version (4 bits):
  *1. Implementations based on this version of the ISAKMP Internet-Draft MUST set the Major Version to 1.
  *2. Implementations based on previous versions of ISAKMP Internet-Drafts MUST set the Major Version to 0.
  *3. Implementations SHOULD never accept packets with a major version number larger than its own.
-Minor Version (4 bits):
  *1. Implementations based on this version of the ISAKMP Internet-Draft MUST set the Minor Version to 0.
  *2. Implementations based on previous versions of ISAKMP Internet-Drafts MUST set the Minor Version to 1.
  *3. Implementations SHOULD never accept packets with a minor version number larger than its own, given the major numbers are identical.
-Flags (1 octet):
  -- E(ncryption Bit) (1 bit)
    *1. It is RECOMMENDED that encryption of communications be done as soon as possible between the peers.
    *2. For all ISAKMP exchanges described in section 4.3, the encryption SHOULD begin after both parties have exchanged Key Exchange payloads.
  --C (ommit Bit) (1 bit):
    *1. However, the value MUST be reset after the Phase 1 negotiation.
    *2. If set (1), the entity which did not set the Commit Bit MUST wait for an Informational Exchange containing a Notify payload (with the CONNECTED Notify Message) from the entity which set the Commit Bit.
  --NOTE:
    *1. If the entity awaiting the Informational Exchange can verify the received message (i.e. Phase 2 SA negotiation message or encrypted traffic), then they MAY consider the SA was established and continue processing.
-Message ID (4 octets):
    *1. During Phase 1 negotiations, the value MUST be set to 0.

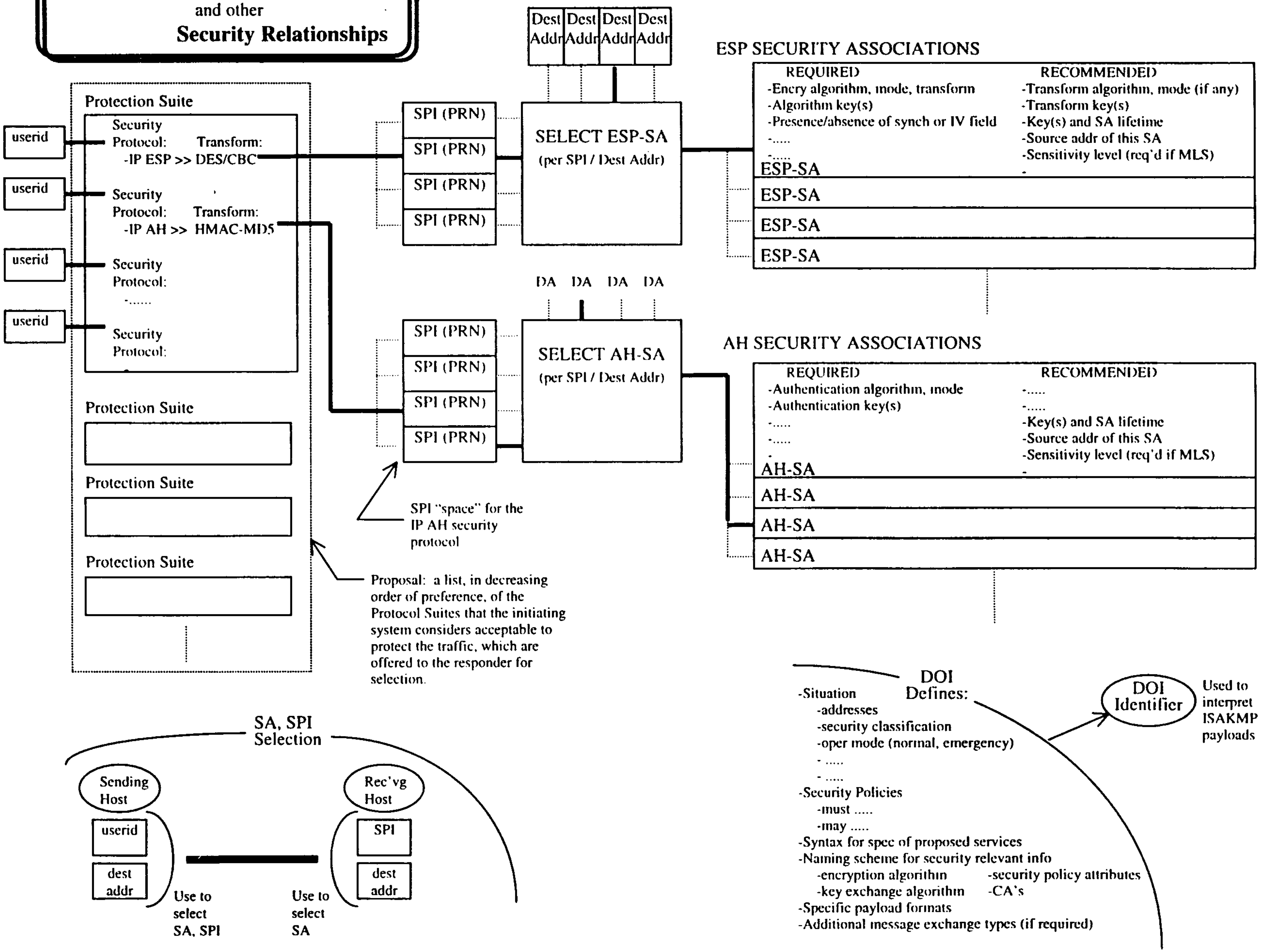
The requirements contained in the following sub-sections in this Section 3 are only counted. The requirements are specific of data format and do not yield insight into the design issues. The 'count' of uppercase requirements in each of these sub-sections is included to indicate the scope of each paragraph. However, the reader is again cautioned that rigor may not have observed in the MUST designations .... been a feature incorporated into the stating of the requirements in the ISAKMP document.

3.4 Security Association Payload - Next Payload	(*1 uppercase requirement)
3.5 Proposal Payload - Next Payload	(*1 uppercase requirement)
3.6 Transform Payload - SA Attributes	(*1 uppercase requirement)
3.7 Key Exchange Payload	(0 uppercase requirements)
3.8 Identification Payload	(0 uppercase requirements)
3.9 Certificate Payload	(*1 uppercase requirements)
3.10 Certificate Request Payload	(*4 uppercase requirements)
3.11 Hash Payload	(0 uppercase requirements)
3.12 Signature Payload	(0)
3.13 Nonce Payload	(0)
3.14 Notification Payload	(0)
3.15 Delete Payload	(*2)
-4. ISAKMP Exchanges	(*1)
4.1 Security Association Establishment	(*11)
4.1.1 Security Association Establishment Examples	(*2)
4.2 Security Association Modification	(*3)

4.3 ISAKMP Exchange Types	(*3)	
4.3.1 Notation	(*4)	
4.4 Base exchange	(*2)	
4.5 Identity Protection Exchange		(0)
4.6 Authentication Only Exchange	(*2)	
4.7 Aggressive Exchange		(*2)
4.8 Informational Exchange	(*1)	
-5. ISAKMP Payload Processing		
5.1 General Message Processing	(*1)	
5.2 ISAKMP Header Processing	(*6)	
5.3 Generic Payload Header Processing	(*2)	
5.4 Security Association Payload Processing	(*4)	
5.4.1 Proposal Payload Processing	(*4)	
5.4.2 Transform Payload Processing	(*3)	
5.5 Key Exchange Payload Processing	(*2)	
5.6 Identification Payload Processing	(*3)	
5.7 Certificate Payload processing	(*4)	
5.8 Certificate Request Payload Processing	(*6)	
5.9 Hash Payload Processing	(*4)	
5.10 Signature Payload Processing	(*4)	
5.11 Nonce Payload Processing	(*2)	
5.12 Notification Payload Processing	(*5)	
5.13 Delete Payload Processing	(*9)	
-6. Conclusions		(0)
-A. ISAKMP Security Association Attributes		
A.1 Background/Rationale	(*1)	
A.2 Assigned Values for the Internet IP Security DOI		(0)
A.2.1 Internet IP Security DOI Assigned Value		(0)
A.2.2 Supported Security Protocols	(*1)	
-B. Defining a New Domain of Interpretation	(*1)	
--- Security Considerations		(0)

7.5

ISAKMP: Security Associations and other Security Relationships



SANDERS
A Lockheed Martin Company

SHARE*HPSC Network Simulation Plan

Rev A

July 22, 1996

This document was developed under the Secure Heterogeneous Application Runtime Environment for High Performance Scalable Computing (SHARE*HPSC) program.

Document # PUBS-96-C35-W

SHARE*HPSC
Network Simulation Plan

Rev A

July 22, 1996

Approved by:	For:	Date:
Jeff Smith	Sanders	07/18/96
Greg Byrd	MCNC	06/24/96
Robert George	MSU	07/22/96

Introduction

This document describes the simulation approach being developed in support of the Secure Heterogeneous Application Runtime Environment (SHARE) program.

The SHARE program includes a series of demonstrations using a physical network with three heterogeneous nodes: a Sun workstation, a PowerPC computer, and a High-Performance Scalable Computing System (HPSCS). This physical network is shown in Figure 1. It will be used to demonstrate various SHARE features, including host-specific versions of code supporting the Message Passing Interface (MPI) and MessageWay standards.

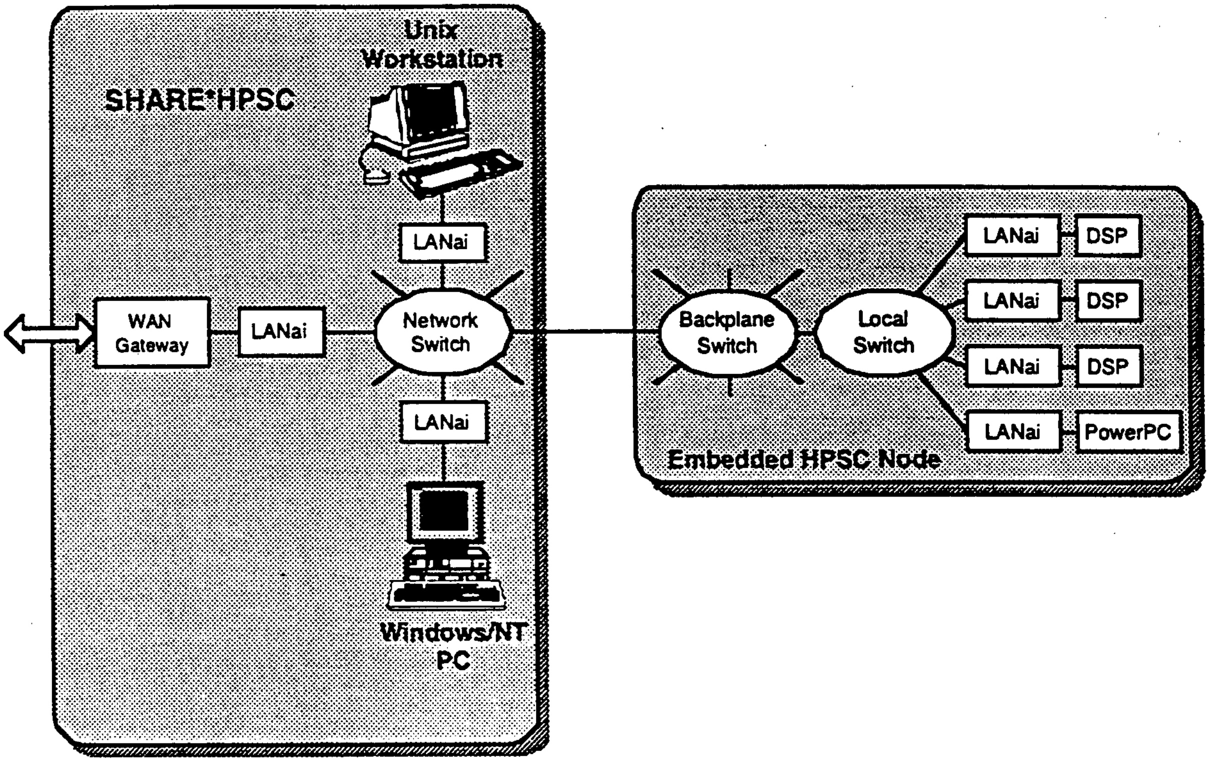


Figure 1: SHARE Physical Network

A simulation testbed will be developed to extend the SHARE demonstrations beyond the limits of the physical network. The simulation testbed should be especially helpful in modeling missing hardware elements and in scaling the network to large numbers of nodes. This simulation testbed is shown in Figure 2. It models the full SHARE architecture, including several System Area Networks (SANs) interconnected over a public network using Secure MessageWay (MsgWay) Routers. The simulation testbed will be used to evaluate alternate hardware/software configurations, measure performance limits, experiment with network tuning, and explore other features of interest. The remainder of this document describes this simulation testbed.

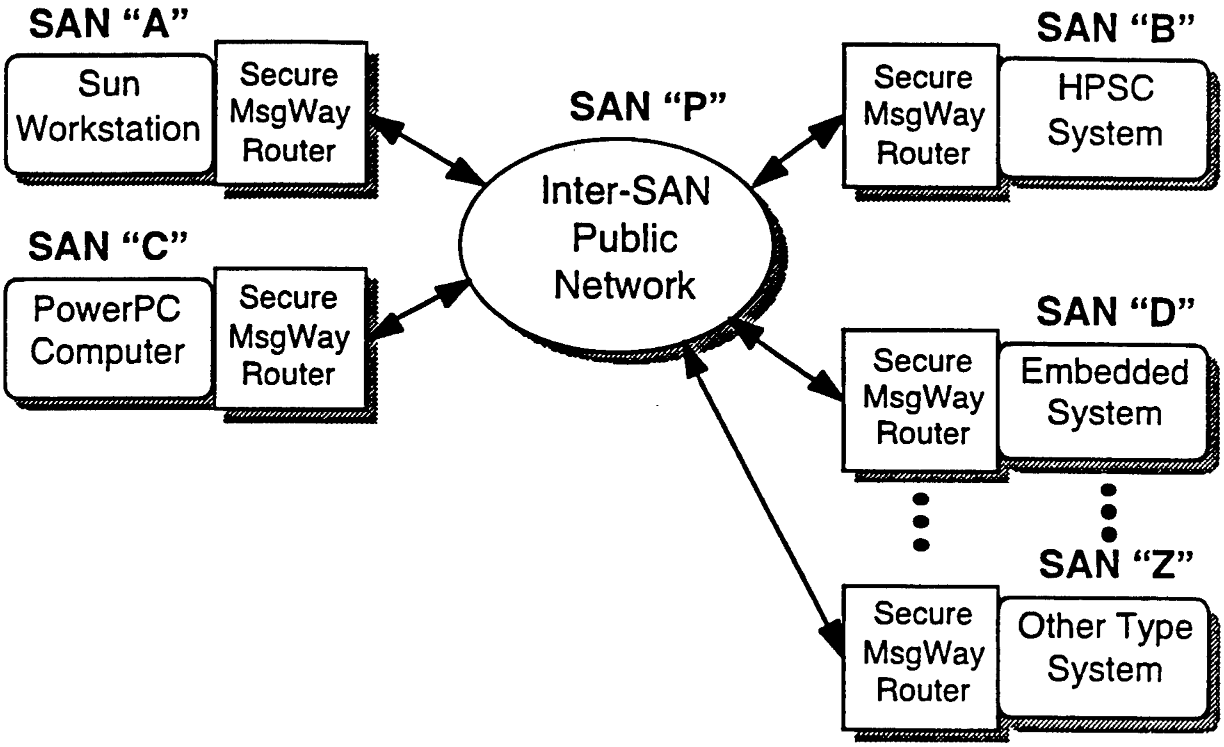


Figure 2: SHARE Simulation Testbed

Simulation Testbed Design

The simulation testbed will be developed as a discrete-event simulator of the SHARE architecture. It will include all the elements of the physical demo network plus various extensions. It will consist of multiple SANs connected by a MessageWay network; models of the Myrinet LANai network interface chip, network, and protocol stack (MessageWay and Myrinet); individual important hardware and software blocks; data flow and timing between models; and transaction scenarios. It will be used as a formal mechanism to verify completeness of interface definitions, verify timing and data flow, verify overall performance of the network, and explore variations in the architecture.

Design of the simulation testbed will be based on the multi-computer Ptolemy work developed on the RASSP and HPSC programs. Higher-level features (processor instruction set, operating system, and MPI code) will be modeled as simple delays, rather than in detailed function. All model features and simulation parameters will be maintained at the unclassified level to avoid the restrictions of a classified project. The simulation testbed will include instrumentation so that data can be easily obtained (and errors analyzed) from the various simulation runs. It shall include the ability to drive the simulation from a stimulus source that provides statistical variation in packet handling.

For those areas where more detailed modeling is desired (such as the crypto module), a hierarchical approach will be taken. A first model will include the basic function, such that it can be used in network simulations for proof-of-concept and system debugging. A second model will include functional details, such that architectural variants can be traded off. The two models will be compatible, such that the more detailed model can be inserted into a simulation in place of the simple model, if desired.

Possible Tests

Specific tests that are planned to be run on the simulation testbed will be a subset of the following list:

- Verify the simulation model by reproducing results from the physical network
- See how the network performs on a typical application (F-22, AEGIS, etc.)
- Try alternate network configurations (topologies)
- Measure when the network breaks under load (nodes, traffic, etc.)
- Show that the network can scale to a large size
- Move the crypto module location and evaluate the network impact
- Show the performance difference between software and hardware cryptography
- Explore multi-level security (MLS) operation of the network
- Insert other node types into the network
- Explore MessageWay packet features (length, distribution, etc.)
- Evaluate alternate MessageWay proposed features
- Fine-tune application performance by load balancing the network
- Optimize application partitioning among network hosts
- Verify component interfaces
- Correct SHARE system requirements
- Explore network reliability (failure and recovery modes)

Application Mapping

One of the tests that could be run on the simulation testbed is to see how the network performs on a typical application. This involves mapping the application onto a set of nodes and then modeling the nodes and network on the simulation testbed. In order to keep the amount of work involved in this test within realizable bounds, it is important that we get a customer (such as the AEGIS group) to work with us to help with the mapping. The mapping should be as simple as possible and should be removed sufficiently from the real application to be declassified. This is an area where it may make sense to seek follow-on funding to complete the work.

Crypto Module Simulation

The simulation of the crypto module is of particular importance because the SHARE physical network will have limited hardware cryptographic capability. The simulation testbed will be used to demonstrate the crypto module function and to experiment with variations in the crypto module design and its location within the network. A first simple model of the crypto module will be developed to enable network tests with multiple secure SANs. A second more complex model will also be developed to evaluate variations in the crypto module implementation.

The secure network concept of SHARE is shown in Figure 3 below. Two separate secure SANs (shown in the lower left and upper right of the Figure) are interconnected over an intervening public network to form a single virtual SAN. The new virtual SAN is made secure, despite the insecure intervening connection, by the use of an encrypting MessageWay router at the boundary of each separate secure SAN.

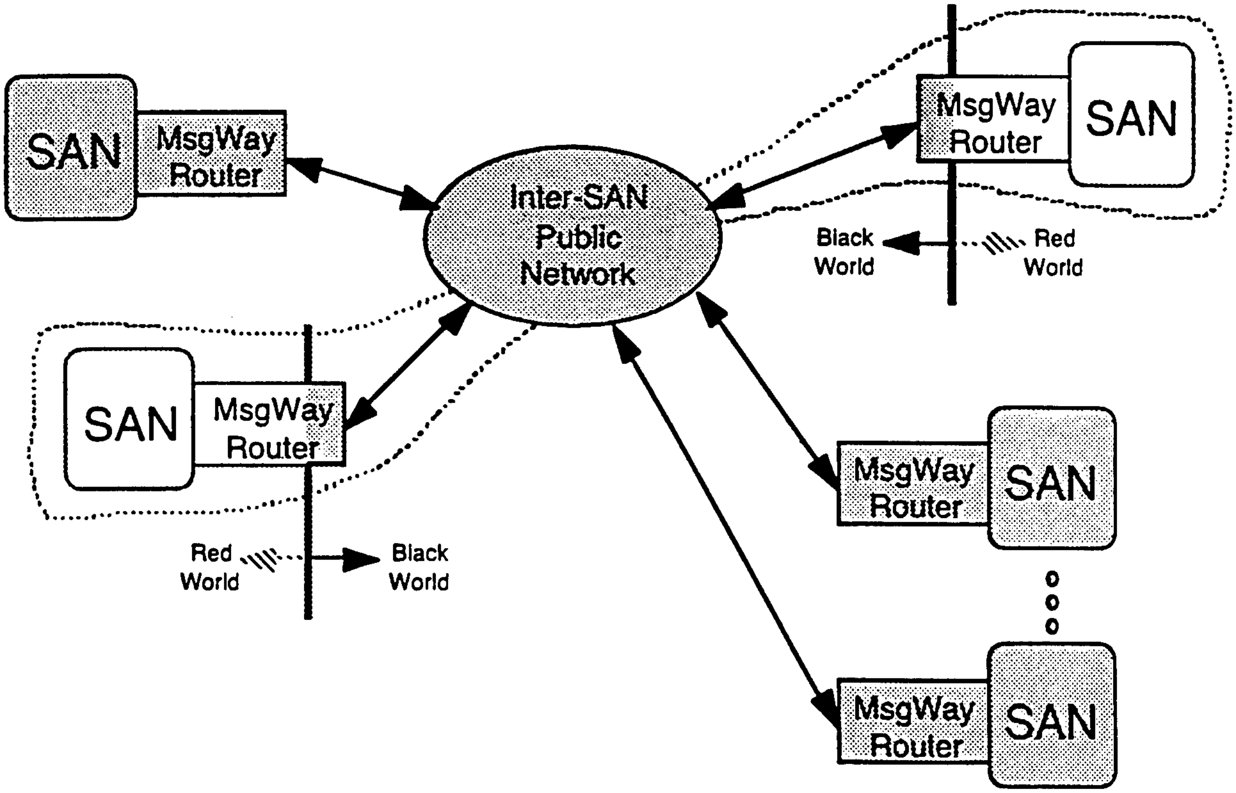


Figure 3: Secure Network Concept

The architecture of the encrypting MessageWay router is shown in Figure 4 below. The router is positioned as a gateway between the local SAN and the inter-SAN network, with the embedded Crypto Module providing privacy and authentication between remote secure SANs. The local SAN network (to the left in the figure) operates at a secure level and is physically protected to that level according to its local security requirements. The public network (to the right in the figure) operates at an insecure level and cannot be trusted to protect unencrypted data packets sent to and from the local SAN network._ The "Red" world (to the left) and the "Black" world (to the right) are separated by the Crypto Module. Outgoing packets from the local SAN are encrypted by the Crypto Module; incoming packets are decrypted and authenticated.

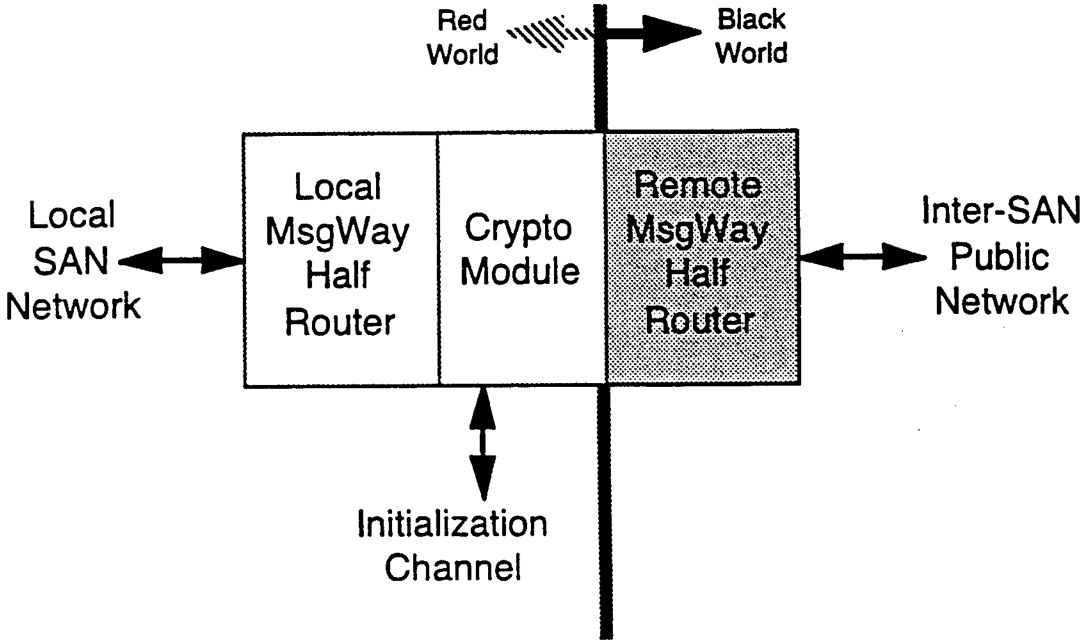
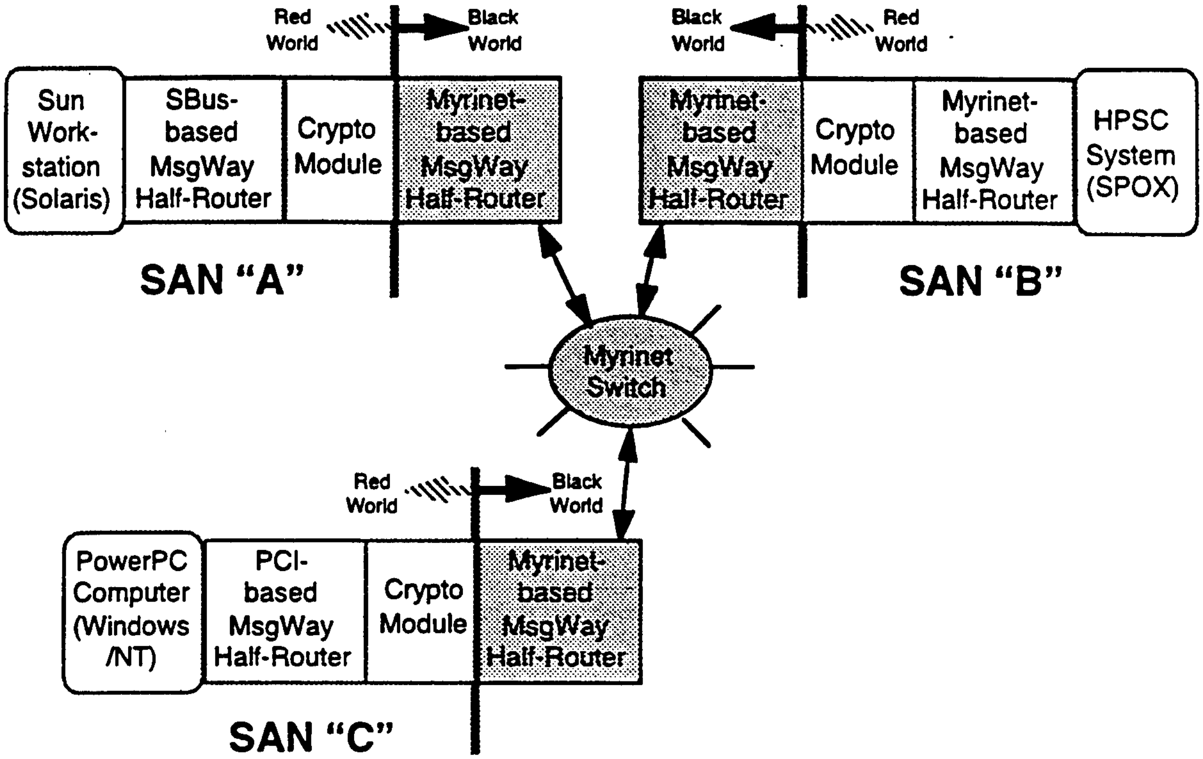


Figure 4: 'MessageWay Encrypting Router

Figure 5 below shows the secure network concept of Figure 3 implemented using the MessageWay encrypting router of Figure 4. This is the base network security architecture that will be modeled on the simulation testbed.



Red World
Black World
Sun Work-station (Solaris)
SBus-based MsgWay Half-Router
Crypto Module
Myrinet-based MsgWay Half-Router
SAN "A"
Black World
Red World
Myrinet-based MsgWay Half-Router
Crypto Module
Myrinet-based MsgWay Half-Router
HPSC System (SPOX)
SAN "B"
Myrinet Switch
Red World
Black World
PowerPC Computer (Windows /NT)
PCI-based MsgWay Half-Router
Crypto Module
Myrinet-based MsgWay Half-Router
SAN "C"

Figure 5: Secure Network Using a Router

SHARE•HPSC

Key Management Plan

Rev –1
7 Nov, 1997

This document was developed under the Secure Heterogeneous Application Runtime Environment for High Performance Scalable Computing (SHARE•HPSC) Program.

SHARE•HPSC

Key Management Plan

Rev –1
7 Nov, 1997

Written By:		Date:
Charles Henter	L–3 Communications	11/__/97
Approved By:		Date:
Jeff Smith	Sanders	__/__/97
Robert George	MSU	__/__/97
Jack Brizek	L–3 Communications	__/__/97

Revision History, SHARE Key Management Plan:

Rev	Rev	Rev	
Level	Date	Author	Nature of Change
–	12/17/96	Lockheed Martin Camden, NJ	–Original Draft
–1	11/7/97	L–3 Communications Camden, NJ	–Modifications and updates, including SHARE team comments (indicated by rev bars). See pgs 3, 16–20, 35–36. –Correction of typos, spelling, and para numbers, and some grammatical improvements (no rev bars).

TABLE OF CONTENTS

1 KEY MANAGEMENT PLAN DEVELOPMENT

1 .1 SCOPE

This document presents the Key Management Plan (KMP) for the SHARE•HPSC* network environment. The Secure Router (SR) is a basic building block in the SHARE network architecture and serves as the interface between a local SAN (trusted, authenticated) and the untrusted public network which provides the data transport to/from the other SRs. A Cryptographic Module (CM) is the component of the SR which performs high speed key agile data encryption and decryption, authentication, and other security functions as described in References 1 through 3. All traffic entering/leaving the SAN from/to the public network is processed by the CM. Additional network agents such as Certificate Authorities and Network Managers supply the required allied network security and management services.

This KMP is the initial SHARE key management document created and conceptually exists at Tier 3 of the government's EKMS/MISSI key management infrastructure. A detailed description of the EKMS/ MISSI infrastructure, and of the SHARE network's design details, exceed this document's scope However, all network concepts are presented herein where they are relevant to key management and the design of the CM.

Section 1 provides background discussion of factors considered in the development of this KMP. Section 2 contains the eleven formal sections required of a Type 1 cryptographic equipment as specified by a typical Data Item Description (DID). Although this KMP is a formal document which uses the DID instructions as a template, substantial discussion has been added to provide the rationale for the key management approaches selected. Sections 3, 4, and 5 contain Acronyms, Definitions, and References which may help this document's readability. Section 6 presents the KMP "Preparation Instructions" as extracted from the DID.

Every effort has been made to present this KMP in a logical and orderly flow. However, there are instances where it has been necessary to introduce a new term without developing its background. In those cases it was decided that to do otherwise would dilute and misdirect the current discussion. The reader is asked to persevere until the background for the new term is presented.

* Secure Heterogeneous Application Runtime Environment for High Performance Scalable Computing

1 .2 FUNDAMENTAL REQUIREMENTS

The fundamental issues from which key management (KM) requirements are derived are listed here. This is the starting point in the derivation of a 'top down' KMP, and the following table provides in barest form an overview of these requirements and the SHARE solutions. Discussions of KM tradeoffs and related SHARE security implementations is provided throughout the body of this document.

KEY REQUIREMENT	KEY RELATED PROBLEM	SHARE KM SOLUTION
–Generation of good keys	–Generation not trivial, especially for public key cryptography –Assurance of strong keys –Symmetric keys easier to generate	–Adopt government EKMS infrastructure (the Central Facility supplies all public/private key pairs).
–Control of keying material over the life of the keys	–Prevent unintended or unauthorized disclosure	–Keys physically and electronically secured (encrypted), tamperproofed, alarmed, and zeroizable –compromised key list enforced

	–Prevent modification	–Keys physically secured, alarmed and cryptographically authenticated
	–Prevent substitution, insertion, replay, deletion	–Keys physically secured, alarmed and cryptographically authenticated in conjunction with a time stamp
–Secure distribution of keys	–Keys transported in the public domain –Must support interoperability between SHARE network elements, i.e. secure routers	–Primitive distribution of split "master keys" –All subsequent key distribution is black –Automatic (network) or manual (DTD) electronic distribution of keys
–Insure integrity of keys and the Key Management Facility (KMF)	–Insure that the KM process is secure, i.e. the keys, the KMF, and the associated procedural controls	–Proven access and procedural controls for the public/private key pairs is provided by EKMS infrastructure –Symmetric keys tested to insure strong keys –Continually test/verify the entire KM process via the provided Monitor and Audit functions
–Recovery in the event of a failure	–Maintain level of protection when integrity of a key or the KM process is compromised	–Equipment (CM/SR) and network alarms and alarm responses for detection and protection –Audit and analyses for identification and correction

1 .3 KEY MANAGEMENT – GENERALIZED TECHNIQUES

A *simple* key management scheme includes (1) the generation of keys in a central location, (2) the manual delivery and installation of the keys to each of the using equipments, (3) similarly delivering and installing key updates as required by the key lifetimes, and (4) providing a protected surrounding environment to insure key integrity and confidentially. Such simplified systems are cumbersome to the user and weak in security. Additional KM functionality and complexity are necessary to provide equipments which do not operationally burden the users and which ensure the desired degree of data security even under worst case threat scenarios.

A fully developed KMP includes a key generation and distribution heirarchy and the associated facilities and equipments to provide the required KM services. The KM services may be grouped into the following categories:

1 . –key generation and selection,

2 . –key availability at the user locations (key distribution, certification, and authentication),

3 . –key destruction.

A fourth category of service must also be in place at all of the above stages:

4 . –key confidentiality.

Items 1 and 2 are expanded in the following paragraphs. All items are expanded throughout this KMP.

1 .3 .1 Key Generation and Selection – Local vs Central

As discussed later, each SHARE CM is identified, certified, and authenticated via its public/private key pair. Such key pairs and their certificates are generated and supplied via the government's EKMS infrastructure. The infeasiblility of key factorization is thereby assured, key dimensions are properly selected, key duplication is prevented, and strong keys are assured. These key pairs are used in the session (symmetric) key establishment process during the call set up between the originating CM and the recipient CM.

Session keys and related key material, e.g. IVs, CVs, sequence numbers, etc, are generated locally (see Section 1 .6 , Derivation of the SHARE KM Approach , for rationale). The symmetric keys are generated from a random process such as provided in hardware by a wide band noise source, or in firmware by a cryptographically secure pseudo–random–bit generator.

1 .3 .2 Key Availability – Manual vs Electronic Distribution

A basic choice to be made is between manual or electronic distribution of key. Manual key distribution is simple and unencumbered, and is most feasible where there are a small number of user sites (SRs) and/ or a low rate of required key replacement. Note that independent of whether manual or electronic distribution is selected, the distribution of the "first master key" must be by primitive/manual means. Subsequently, electronic key distribution may be used, and the keys may be distributed as Black keys, to be decrypted by the "master key" before placing in Red storage in the CM. The "1st master key" may be transported by special courier and may be in the form of split keys (split knowledge) to further protect confidentiality, or it may be wired in during the latter stages of fabrication of the CM. Master key storage for manual conveyance may take the form of a (zeroizable) "key gun" or an electronic equivalent which inserts key directly into the CM. All keys, including the "master key", must be changed at regular and unpredictable intervals to limit the data exposed to an attacker in the event a key is compromised. Key lifetimes are thereby much shorter than the perceived 'key cracking' interval.

1 .3 .3 Key Distribution by Public vs Symmetric Key Cryptography

Where electronic key distribution is selected, there is the option of distribution by symmetric or public key cryptography. Symmetric key distribution has the advantage of speed, i.e. symmetric key lengths are shorter, and public/private encryption and decryption is much slower than the equivalent symmetric key method for the same security (~1000x). The typical method for secure key distribution via symmetric keys is that employed by Kerberos. This method employs an authentication server (Kerberos) and a Ticket Granting Server (TGS). The authentication server provides access control and is a trusted third party arbiter which allows the originator to access/communicate with prescribed services on the network. After authentication, the originator requests a ticket to use the network service from the TGS. The TGS ticket authenticates the originator to the network service. In this method, authentication exists via knowledge of the other's key, and via the presentation of data encrypted in that shared key.

Key distribution by public key cryptography is appropriate for large networks with a large number of user nodes. This is the SHARE case where network scalability is a prime requirement, i.e. where a large number of secure routers require a large number of keys. Since key distribution is essentially a network signalling service performed in parallel with data transfer, data transfer performance (bandwidth, latency) is not directly affected. An exception to this is for the case where the network is overloaded and any time allotted for signalling activity reduces the data bandwidth.

SHARE minimizes the chance for such bandwidth degradation by assigning the responsibility for symmetric key material generation to the CM originating the call. Once the parameters for the virtual channel are negotiated (during the 'first call' set up), these parameters are stored and do not have to be re–established until and unless a CM has unusual traffic requirements. Likewise it is not necessary to poll a Certificate Authority (CA) for the party's signed certificate every time that traffic is requested between two SANs. Again, the exception to this is only when a given CM's network loading exceeds its large 'stored' capability, and new keys and certificates are required.

1 .4 KEY MANAGEMENT – A HEIRARCHY DISCUSSION

The government's key management heirarchy is the EKMS, or Electronic Key Management System. There is also an effort to implement technology upgrades to this heirarchy via MISSI, the Multilevel Information System Security Initiative.

The EKMS heirarchy is multi–tiered to allow for key management authority and responsibility to be partitioned, and for the keys to be managed and accounted for accordingly. The CM addressed in this KMP is intended to be certified as a Type 1 device which handles data at classification levels of Top Secret and above and below, including that associated with compartmentalized SANs requiring special access control. As an adjunct to this certification, the government's EKMS is to supply critical key material required for the operation of the Type 1 and 2 CMs. In brief, EKMS is asked to supply the asymmetric public/private key pairs and certificates that identify and authenticate each secure router's CM. These keys are used for communication privacy and authentication in call set up negotiations (SCID negotiations) between CMs. These negotiations include the transfer of symmetric session keys and related key material protected by the public keys. The specific types and amounts of key material required are detailed in Section 2 .3 .2 , Number Of Keys Required .

The advantage to SHARE which results from using EKMS is that the necessary KM infrastructure exists to ensure strong public keys, key integrity, secure distribution, certification, authentication and accountability. And the associated rules, regulations, and procedures are in place and well defined.

EKMS separates key management authority into Tiers 0 through 3. This CM KMP exists at Tier 3. The following definitions are provided to give a brief explanation of the tier structure:

Tier 0: The Central Facility (CF) – The CF is the NSA facility which provides centralized KM services, i.e. the central location from which key material is ordered and generated. Key material may be manually distributed to Tier 1 elements, e.g. via magnetic disk.

Tier 1: Service Accounts Facility, i.e. the Depot (Army, Navy, Air Force, etc) – This tier is a facility comprised of large computers which generates, manages, and distributes key to its participating elements. Initialization, system privileges, and record centralization are additional services provided by this facility. MISSI LMD/KP equipments may interface the facility to Tier 2 elements.

Tier 2: Comsec Accounts – Tier 2 devices will be primarily LMD/KP workstations which provide for the automated management of Comsec key material. Key material may be loaded into and manually distributed to Tier 3 elements via electronic Data Transfer Devices (DTDs). The DTD is an electronic automated fill device used for key transport and load into the using COMSEC equipments.

Tier 3: Comsec End Users – The SHARE CM is the Comsec end user, and contains a DTD port for the loading of black public key material generated and supplied via this heirarchy. The SHARE network design also provides for a network key distribution capability.

1 .5 POSSIBLE SHARE KM APPROACHES – A SYSTEMS DISCUSSION

In the SHARE case, there are three possible systems on which to base a key management plan:

1 . –the current Electronic Key Management System (EKMS) used by NSA for the management of key material for use by the US government and military,

2 . –the EKMS including ongoing upgrades as provided by the Multilevel Information System Security Initiative (MISSI) being developed by NSA, or

3 . –a SHARE self contained, self sufficient key management system.

After a few definitions, it is shown that the SHARE CM uses a combination of these three systems.

(1) EKMS: EKMS is the unified system used by the government for the electronic generation, dissemination, and management of key material for its subscribers. At the using equipment level, i.e. at the SHARE CM level, keys are loaded into the using cryptographic equipment via a Data Transfer Device (DTD). The DTD in turn has been loaded with keys under the auspices of a COMSEC Account (CSAC) as managed by a Central Office of Record (COR). The keys passed to the end using equipment are controlled via a COMSEC Material Control System (CMCS).

(2) MISSI: MISSI is the effort to provide multiple security mechanisms/products to protect DoD's future Defense Information Infrastructure (DII). Firewalls, in–line network encryptors (INEs), and Fortezza and Fortezza+ cards are some of the MISSI building block products. A primary way of loading keys into using equipments is via a Fortezza card containing appropriate public key protection mechanisms. The EKMS system of key supply and management will incorporate these and future MISSI developed components as they become available.

(3) SHARE Self Contained: A self contained KM system which does not interact with EKMS can be design optimized for data related performance in the SHARE environment, i.e. high data rate, low latency. However, EKMS may be viewed as being optimized for KM security, and MISSI will make EKMS more manageable and therefore more secure. Note that a SHARE self contained, self sufficient key management system is required in those cases where the network environment is to be used in purely commercial, non DoD entities.

This KMP is developed with the intent of the final equipment design implementation being compliant with a class B2, "Mandatory–Structured Protection" certification level, i.e. as defined via TCSEC – Trusted Computer Security Evaluation Criteria.

1 .6 DERIVATION OF THE SHARE KM APPROACH

1 .6 .1 A Simple Key Establishment Scheme

Before secure communication can take place between two entities, the keys must be established and compatible security attributes must be assured. A simple key establishment scheme would be for all network CMs to store identical security parameters, including identical key sets, in their memory banks. The key establishment method would then be to negotiate via network protocol (Secure PacketWay) a SCID (Security Context Identifier) to be used as a memory pointer to the keys and other security attributes to be in effect during the data transfer. The number of session keys that the SHARE CM is required to handle with agility has been established as 64k. Header–MAC keys, full_message–MAC keys, and IVs are also associated with each of these 64k session keys. In this simplified scheme the keys would be supplied by the CF at Tier 0, and the keys distributed to the CMs for their use. A first concept is to consider that all CMs have identical 64k key sets in storage. However, this means that an attacker located "inside a secure Secret SAN" (who has access to these keys) only has to test his 64k key set against an encrypted message intended for another SAN, rather than searching/testing the entire key space. If a Top Secret SAN has "read down" access to Secret and/or SBU SANs, then his memory bank of Top Secret, Secret, and SBU keys will give him "attack access" against all of these SANs.

An attacker who resides outside of the secure SAN has no knowledge of the selected 64k keyspace subset and must search/test the entire key space. However, the large amount of traffic available for analysis to the outside attacker, which has been encrypted with this smaller key set, is presumed to provide the additional information which allows him to reduce the cracking time.

Protection can be provided against inside attack by assigning a different key for each possible communicating pair of CMs. For a network of 1000 SANs, 499,500 session keys plus an equivalent number of header–MAC and full–message–MAC keys, and IV vectors, are required. And 1000 different key loads are required, i.e. a different key set for each CM. Alternatively, if the amount of key sets in the net were limited to 64k (65,536), then the net could accommodate only 362 CMs. This simple scheme thus requires the generation and distribution, replacement and destruction, and management of a large amount of key material. Or the network must be limited in the number of CMs (SANs) that can be accommodated. Making this scheme work is thus seen to require unsatisfying choices between security, key set size, and/or the number of accommodated SANs.

1 .6 .2 A Better Scheme – SHARE Key Establishment:

The problems described above are created as a result of using an externally supplied set of session keys, i.e. if all CMs have the same supplied key set, then an inside attack is feasible, and if all CMs have a different supplied key set, then a large amount of key material is required.

To eliminate this problem, symmetric keys are generated within each CM and supplied when required, i.e. not apriori of any CM communication need. Only the public/private key pairs for the networked CMs are supplied and installed before placing the CM/SR on–line. Each CM only requires knowledge of its own public/private key pair and the public key of each of the participating CMs. When a public key is required that is not in the CM's 64k storage, a Certification Authority (CA) located in the SHARE Network Manager is consulted. The CA supplies the needed public key, signed by the CA and authenticating the intended recipient's identity. This process ensures the validity of the public key and thereby prevents a man–in–the–middle attack. A CM which is just coming on–line does not actually require an initial pre–determined public key load of 64k keys, although this is most efficient. Storage of the public keys for the authenticated CMs at the CA, and their distribution only as needed (and acquired as part of each call set up) by the CM, would eliminate the need for a bulk key load from either the DTD or the network. This 'key by key' acquisition method is inherently available in the CM design, albeit at the expense of a longer call set up for all 'first calls'.

The negotiation of a SCID and key set takes place under the protection of the two communicating parties' public and private keys. The symmetric keys to be used for data and MAC encryption, and the IVs, are generated within the source CM and transferred to the destination CM as part of the SCID. The key exchange messages are time stamped to prevent replay. All subsequent data transfers between the two parties then use the established keys and IVs for the assigned key lifetime.

Note that symmetric keys are relatively easy to generate and to test for weakness. Their generation and storage may take place at times of low data activity, and at periodic (random) intervals with the intervals being determined either by the CM or by the Network Manager. Also note that although the number of nets that a CM may communicate with is limited to 64k at any one 'instant', SCID/key establishment at call time allows for communication with an unlimited number of nets, i.e. via the dropping of 'old' SCIDs/keys and the generation/establishment of new ones as required. In this respect, the 64k key set existing in CM memory may be viewed as a cache for the rapid (low latency) establishment of traffic for the more likely destination requests. The 64k key set storage capability is also currently viewed as 'large'. The associated ability to communicate with 64k other DoD SANs without going through the CA is considered adequate for some time to come.

1 .6 .3 Latency Overhead

In the simplified scheme first described in Section 1 .6 .1 , a SCID is negotiated in the open when traffic is requested, and at some cost in set up time. In the SHARE key establishment scheme described above in Section 1 .6 .2 , a SCID negotiation still takes place, but is masked by public key cryptography, and the SCID negotiation includes agreement (transfer) of the symmetric keys to be used. This approach is at a greater time expense because of the longer encryption/decryption times required of the public key data. However, the increase in set up time cost required to process the key exchange does not impact data transmission performance once the flow of data packets begins. And the SCID/key set up costs accrue only during SCID negotiation which may take place only once key lifetime, or at least at a relatively low rate. See Section 2 .3 .2 for a discussion of key lifetime.

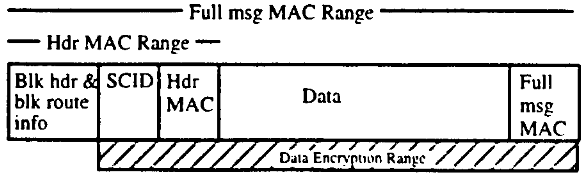
1 .6 .4 Secure PacketWay

Key exchange occurs via a provision in the network packet switching protocol, Secure PacketWay, and is based on the Diffie–Hellman Key Agreement protocol. The Diffie–Hellman protocol is the basis for a number of government based key establishment protocols. The Secure PacketWay Key Exchange Protocol is presently under development.

1 .7 SHARE MESSAGE AUTHENTICATION

Separate comment is offered here on the data packet message structure, the ranges of encryption and message authentication, and considerations which resulted in the authentication approach. The data packet is encrypted by a session key (Type 1 or 2 algorithm) over the range as shown, with the (black) part of the header maintained in the clear to enable the routing of the packet through the public network. Two additional keys are used in the creation of the data packet, i.e. one each for the protection of the two MACs. The MACs are message digests (hashes) generated as non linear functions of the input data. The non linear functions are implemented via algorithms and hardware similar to those of the session encryptor.

Packet/MAC Message Structure



The first (header) MAC is calculated over the first part of the packet, i.e. the portion containing the black source routing information and the negotiated SCID. The SCID contains the negotiated communication attributes including the session key and IVs. On decryption, this MAC allows an early/ quick verification of the integrity of the source routing information, and the ability to terminate the packet if the routing info is faulty.

The final (full message) MAC is placed at the end of the packet and serves to authenticate the integrity of the full message.

The protocols which facilitate this routing are a part of the Secure PacketWay standard.

Considerations which led to the message authentication approach are briefly stated below:
The high rate of data throughput requires a high rate MAC, and there are a limited number of approaches, all in hardware, that can sustain the required speed.

An XOR MAC was considered for SHARE which employs the DES algorithm (and a key) as the means for generating the PR data block required by the approach. The approach is described in Reference 10. Thus a key is required for each MAC so implemented. The key might be the same as the session key, however possible consequences associated with the additional information that may be provided by the duplicate use of the session key have not been established. Also, an all zero key might be used for the XOR MAC and the resultant MAC encrypted normally with the session key.

A second considered method is to generate a hash over the plain text, and encrypt the hash with a key. If this is done, then the encryption must include a means for insuring that the same plain text does not encrypt to the same ciphertext, i.e. via CBC, CFB, OFB, Counter Mode, or Long Cycle Chaining.

A third considered method is to generate a hash over the ciphertext, and separately encrypt the resultant hash. To encrypt the hash, a second key and IV are required. If the hash is not encrypted, then the ciphertext can be altered and a new hash created to match. A message that has been maliciously garbled would then be detected as a valid message.

In the method selected, both the header and full message MACs are calculated via the same algorithm and via hardware similar to that used to encrypt the data itself. The basic design is thus a near replication of the data encryption circuitry. In this case however, the MAC hardware is operated in the CBC or CFB mode. In these modes, MAC circuit output depends on previous input/output and is thus a unique non–linear–encrypted–hash of the plain text which has preceded it. The circuit output at the end of the black routing info and SCID is thus the message digest required. This same MAC circuit continues its calculations to the end of the data packet to yield the full message MAC as shown. Separate keys are used for these MAC calculations to ensure that the MACs do not provide allied information on packet data content.

1 .8 KEY SUPPLY – IMPLEMENTATION DISCUSSION

1 .8 .1 Key Supply – Final Implementation

Key supply for the SHARE CM final implementation is outlined in Section 2 .3 .2 . That section includes a definition of the required memory space for the key material, and whether the key material is self generated within the CM or supplied externally via EKMS. EKMS material may be delivered either by a DTD or transmitted from EKMS over the SHARE network. Network distribution presumes that the EKMS distribution center has connected the necessary SHARE secure router(with CM) to its local SAN.

1 .8 .2 Key Supply – Phase 1 Implementation

The plan for Phase 1 is to fabricate a breadboard CM and to implement the parts of the KMP which are required to demonstrate CM bandwidth, latency, and key agility. To demonstrate these performance parameters, the 64k key memory cache must be in place. Note that each key in the cache is normally associated with a SCID and CV containing the security associations which facilitate the key's use. Although by this definition it is not necessary to locally generate symmetric key material, nor to move data between the memory partitions, it is considered important in Phase 1 to have in place a method of getting a satisfactory quantity of material into the 64k space which allows for rigorous key agility testing. Note that Reference 2, "System Requirements", for Phase 1 designs, simulations, and demonstrations, keys may be manually inserted into a look up table which is accessed by the cryptographic function. In this regard a 'large amount' of key material, with sufficient identity markers to permit a valid test of performance, will be loaded into the memory key space. Since key establishment with the data recipient CM is a Secure PacketWay signalling protocol and is accomplished in parallel with data operations, it does not contribute directly to the stated performance parameters, and will not be implemented. However, key establishment processing and memory size partitions will be identified and key establishment execution times will be estimated and incorporated into the design architecture.

2 FORMAL KEY MANAGEMENT PLAN

A formal Key Management Plan provides a detailed description of the key scheme proposed for a cryptographic system or equipment. It documents in detail how key generated in support of the cryptosystem or equipment will be managed from the time it leaves the point of generation until it is destroyed. The ultimate purpose of the KMP is to provide the Government Office of Key Management sufficient information for system and/or equipment certification, and for determining compatibility and supportability by existing or proposed Key Management Systems. This KMP is intended to satisfy these requirements for the Cryptographic Module (CM) of the SHARE Secure Router (SR).

A second purpose of this KMP is to provide guidance in the design of the CM and its key management features. This KMP and the technology which is implied, address the final design of the CM and its final SHARE network implementation. This KMP is also applicable to CM Phase 1 development, and is developed in parallel and in concert with the Phase 1 CM design.

In a typical government contract specifying the development of a cryptographic equipment, a Data Item Description (DID) is provided which defines the requirements of the Key Management Plan. For this document, a representative DID has been selected, DI–OT–0021, to serve as the basis for KMP design. The eleven subsections formally requested by this DID correspond directly to the eleven subsections which follow. "Preparation Instructions" for these eleven subsections have been extracted from the DID and included for reference in the Appendix in Section 6 . Also, each subsection below is preceded by a digest (italicized) of the DID's "Preparation Instructions" and may be used as a topical summary of the contained discussion. The corresponding ✓ confirms that the requirement has been covered as requested in the DID.

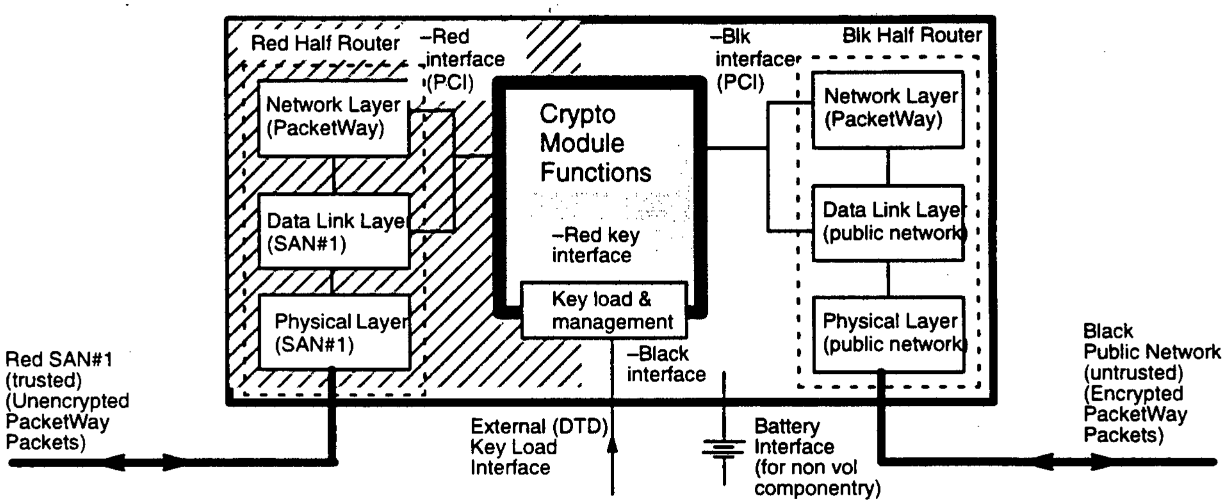
There is some overlap in the sections which follow. For instance, the INTRODUCTORY INFORMATION section requested in the DID overlaps information requested in the COMMUNICATIONS ARCHITECTURE section, which in turn overlaps information requested in the KEYING SCHEME, which also overlaps information requested in the NET STRUCTURE, etc. There has been no attempt to remove this overlap as it has been specifically requested by the DID. This means that some paging 'forward and backward' to referenced sections is necessary to get a complete description of the subject.

2 .1 INTRODUCTORY INFORMATION

✓ *–purpose of the item*
✓ *–identify the item as type 1 or 2*
✓ *–describe the system in which the item will be used*
✓ *–terminology throughout the plan shall be consistent with the National INFOSEC Glossary.*

2 .1 .1 Purpose of Item

The CM is a component of the SHARE Secure Router (SR). The SR is a building block in the SHARE packet network architecture and serves as the interface between a local trusted authenticated SAN and an untrusted public network. Within the SR, the CM resides between the red and black half routers as illustrated in Figure 1. The CM performs high speed key agile data encryption and decryption, authentication, and other functions and security services as described herein and in References 1 through 3 . It implements the Reference Monitor concept and functions as the security kernel which controls all access to the cryptographic services provided by the CM. All traffic entering/leaving the SAN from/to the public network is processed by the CM. A top level block diagram of the functional elements of the CM and the related major signal flows is illustrated in Figure 2. The figure is provided here for insight into the operation of the CM. A complete description of the CM's requirements may be found in Reference 3.



Red Half Router
–Red interface (PCI)
Network Layer (PacketWay)
Data Link Laye (SAN#1)
Physical Laye (SAN#1)
Crypto Module Functions
–Red key interface
Key load & management
–Blk interface (PCI)
Blk Half Router
Network Layer (PacketWay)
Data Link Laye (public network
Physical Layer (public network
–Black interface
Red SAN#1 (trusted) (Unencrypted PacketWay Packets)
External (DTD) Key Load Interface
Battery Interface (for non vol componentry)
Black Public Network (untrusted) (Encrypted PacketWay Packets)

Figure 1: SHARE Secure Router

The Secure Router data throughput capability is 160 MB/s, and 320 MB/s full duplex. The design supports multilevel security, i.e. it supports key agility on a per–message basis and allows the assignment of keys based on security context.

A PCI (Peripheral Component Interconnect) data interface (or other open system standard interface providing the required bandwidth) to the CM is used to provide the high bandwidths required. Two PCI buses are perceived, a 2x32 bit black bus for the untrusted public network side, and a 2x32 bit red bus for the trusted local SAN side. Connections to the CM are shown in Figure 1 from both the data link and network layers in order for the CM to access the plain text and security context information required to process a message packet.

A black external key load interface is also shown in Figure 1. This interface is for the purpose of loading public keys and the associated signed certificates for the authenticated CMs in the SHARE network. In addition, as shown in Figure 2, there is a network key load interface which also provides for this key loading via the network. Symmetric session and MAC keys, and related synchronization vectors, are generated in the local CM itself and transmitted to the called recipient as part of the call set up protocol. The SHARE design concept also includes the provision for loading and storing split CM "master key pairs", including the CM's "first key pair" that allows subsequent key loading to take place in black form. A Crypto Ignition Key (CIK) is used to 'turn on' the CM/SR and place them in an operational mode.

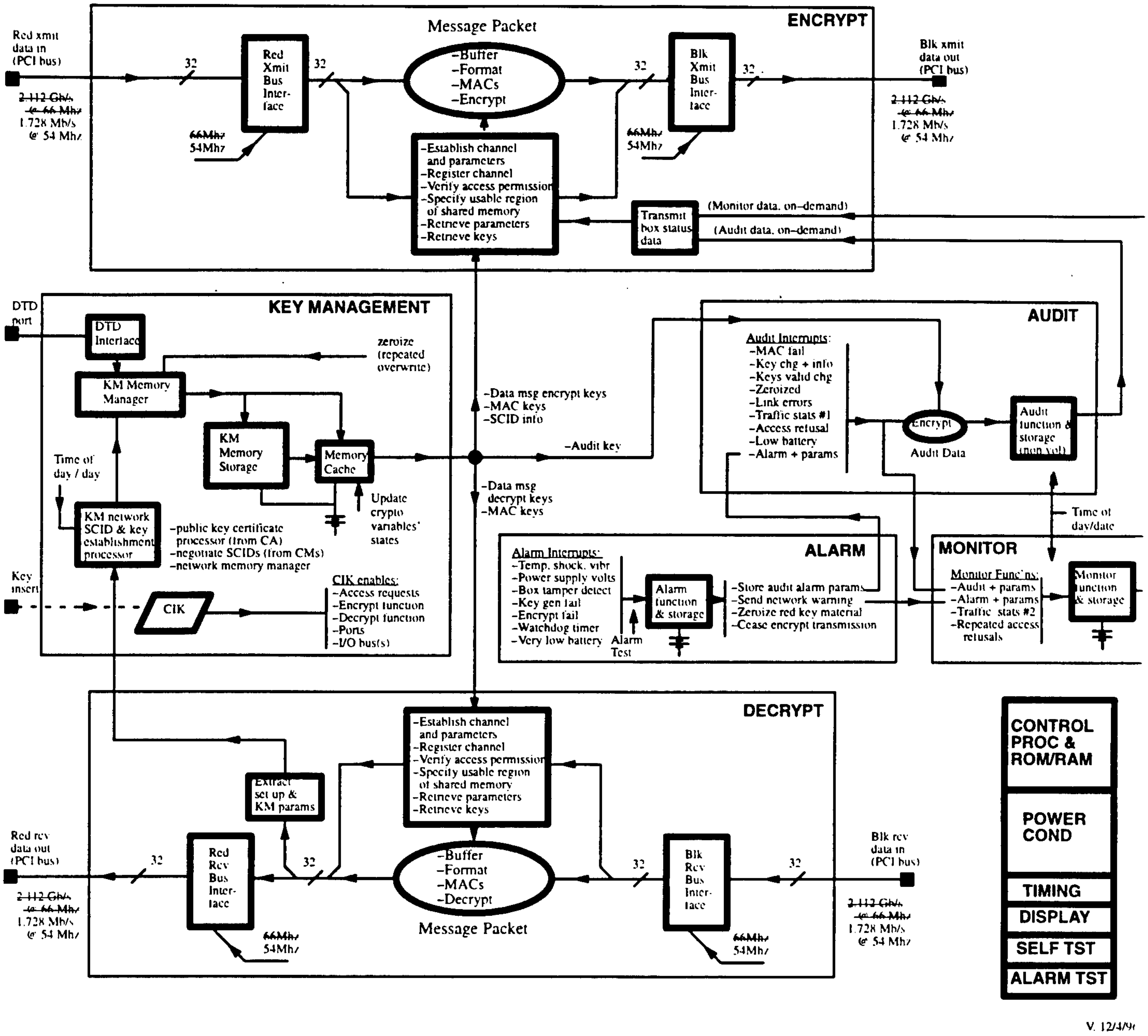


Figure 2: SHARE CM Signal and Functional Summary, Logical Partitioning

2.1.2 Type of Item

The CM design plan is to support both Type 1 and Type 2 requirements.

2.1.3 The System In Which the Item Will Be Used

SHARE•HPSC is a fully integrated network environment that links System Area Networks (SANs) of any design, and provides high speed, low latency, multi–level secure information transfer between the

SAN hosts. SHARE allows authenticated SANs of differing types and security levels to communicate at 1.28 Gb/s rates (2.56 Gb/s full duplex) over a public network.

SHARE is an open system utilizing Secure PacketWay as the network packet switching standard to connect and accomplish information transfer between the high performance local SANs. SANs may transmit and receive data from peer SANs of like security level, and SANs of higher security level may read data from lower level SANs, i.e. 'read down' concept. These data transfers are permitted when they are consistent with SAN security policy, access controls, authentication levels, and specifically granted security authorizations. A SAN host node can range from a single workstation or personal computer to an embedded special purpose scalable multiprocessor.

2 .2 COMMUNICATIONS ARCHITECTURE

✓ *–comm arch shall specify the comm requirements (voice, data).*

✓ *–describe the comm scenario (LAN, broadcast, pt to pt, telephone, etc).*

✓ *–specify the min and max net sizes.*

SHARE is designed to transfer data via a network of point to point links that can be configured as any of several topologies, i.e. tree, star, mesh, etc. An example SHARE network structure is illustrated in Figure 3. Local SANs of various security levels are connected via Secure Routers (SRs) to the high speed public network. A public network switch, e.g. Myrinet, provides the physical and data link layers, and has a compatible data transmission bandwidth of 160 MB/s (1.28 Gb/s) and a full duplex transmit/receive bandwidth of 320 MB/s (2.56 Gb/s). For the classified SANs, the associated SRs operate via a Secure–PacketWay network layer protocol. Standard routers, or SHARE/PacketWay routers without the cryptographic module, may connect unauthenticated SANs. The SHARE Network Manager and the Network Operating System facilitate network operations via associated controls and displays. The SHARE Network Manager incorporates the functions of network key management, certificate authority, mail list agent, directory service agent, audit manager, and other functions as discussed in Reference 2.

The minimum net size is two SANs, i.e. there must be two SANs in order for there to be communication. The maximum net size is unlimited. Each CM contains a 64k cache of 'previously negotiated' SCIDs (session keys, security context IDs, security associated data). The SCIDs are stored in fast cache RAM for the support of key agile packet data encryption. The CM provides max data rate and minimum latency performance for the 64k cached SANs. 'New' SANs are added by replacing an 'old' cache location with the newly negotiated SAN's SCID.

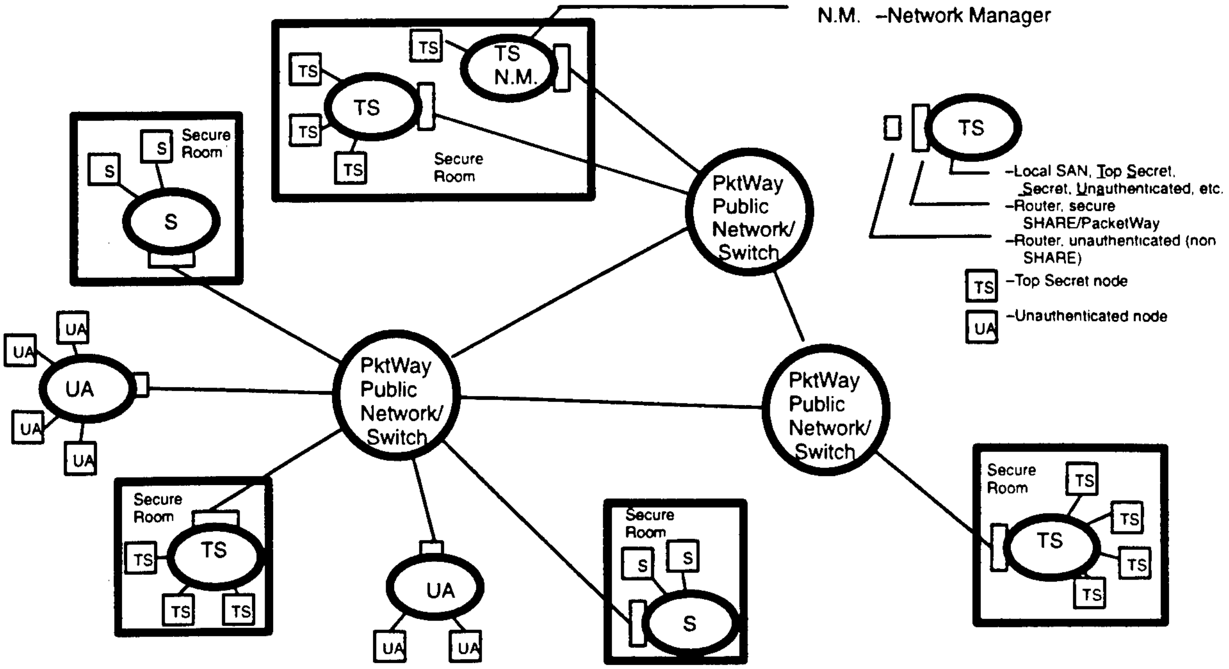

N.M. –Network Manager
TS
TS N.M.
TS
TS
TS
TS
TS
Secure Room
S
S
Secure Room
S
PktWay Public Network/ Switch
PktWay Public Network/ Switch
PktWay Public Network/ Switch
UA
UA
UA
UA
UA
UA
UA
TS
–Local SAN, Top Secret, Secret, Unauthenticated, etc.
–Router, secure SHARE/PacketWay
–Router, unauthenticated (non SHARE)
TS –Top Secret node
UA –Unauthenticated node
Secure Room
TS
TS
TS
TS
UA
Secure Room
S
S
S
Secure Room
TS
TS
TS
TS
TS

Figure 3: An Example SHARE•HPSC Network Structure

2 .3 KEYING SCHEME

✓ *1–define keying scheme in detail*

✓ *2–address the key requirements (how many different keys are used by the item).*

✓ *3–identify who will supply the key to the user.*

✓ *4–specify the distribution medium (physical, electronic, other).*

✓ *5–identify when key is RED and when key is BLACK.*

✓ *6–identify other components required to enable the item to operate securely (fill device, CIK, Smart Key, etc).*

✓ *7–identify the above component's source (if its not part of the system proposed).*

2 .3 .1 Description of Keying Scheme

✓ *1–define keying scheme in detail*

✓ *4–specify the distribution medium (physical, electronic, other).*

✓ *5–identify when key is RED and when key is BLACK – IN THE CM.*

✓ *5–identify when key is RED and when key is BLACK. – IN THE KEY SUPPLY NET*

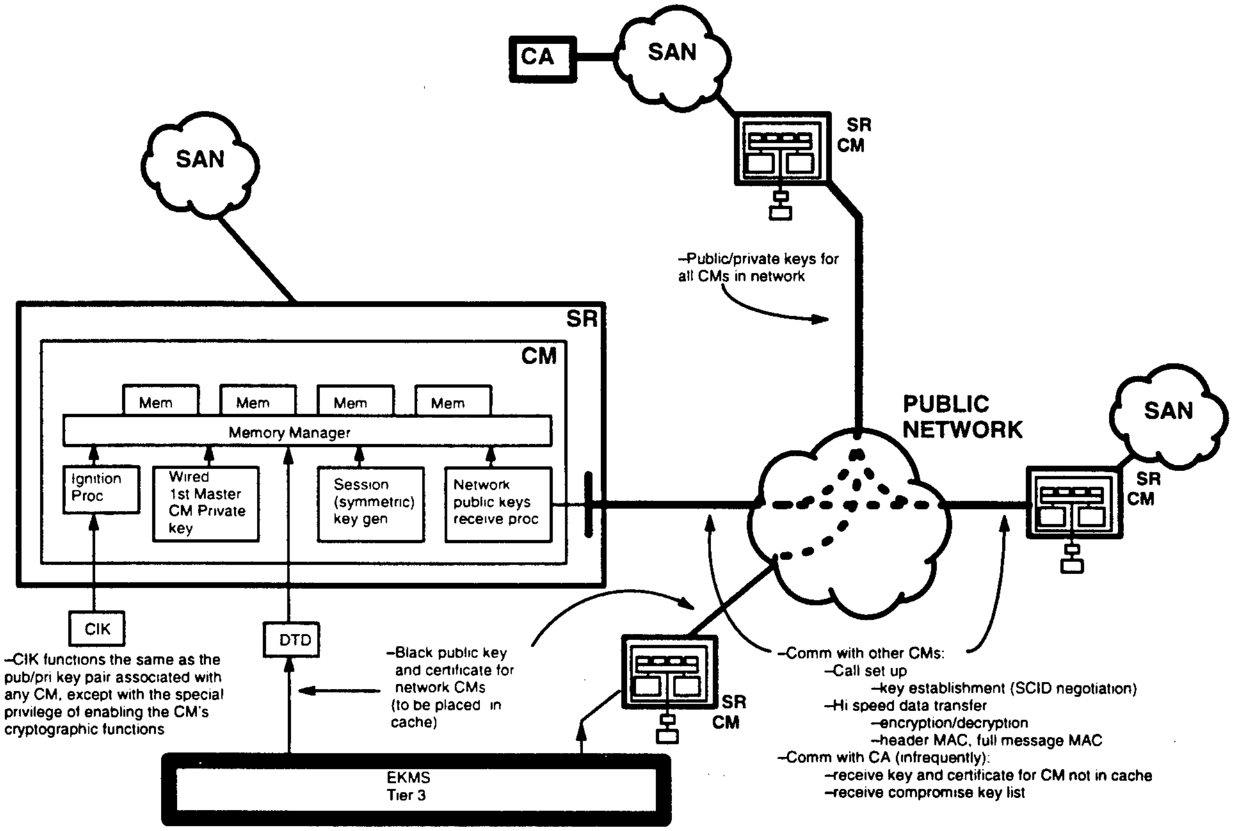
2 .3 .1 .1 Keying Scheme Overview

An overview of the key generation and distribution process from a network viewpoint is illustrated in Figure 4 and described below. An overview of the key memory management and usage process is also summarized below with the aid of Figure 5.

2 .3 .1 .1 .1 Overview – Network Key Generation and Distribution Summary

As illustrated in Figure 4, the public keys and associated certificates (of the other SR/CMs in the network) are provided by EKMS/Tier 3 and may be loaded (1) from the SR's DTD port (see bottom of the SR block), or (2) via the network interface as illustrated by the path from EKMS/Tier 3, to the associated SR/CM, through the Black Public Network, to the subject SR/CM. The CM memory manager directs the storage, processing, and flow of these keys through the CM as they are decrypted and prepared for use in communication with other SR/CMs. All participating SANs may similarly and securely receive the public keys and associated certificates for the SR/CMs in the network.

Black keys so received via the DTD or network port are temporarily placed into black memory. At the conclusion of the key load, the keys are decrypted and authenticated and placed in red 'pending memory' as described in the following.



–Public/private keys for all CMs in network

–CIK functions the same as the pub/pri key pair associated with any CM, except with the special privilege of enabling the CM's cryptographic functions

–Black public key and certificate for network CMs (to be placed in cache)

–Comm with other CMs:
  –Call set up
    –key establishment (SCID negotiation)
  –Hi speed data transfer
    –encryption/decryption
    –header MAC, full message MAC
–Comm with CA (infrequently):
  –receive key and certificate for CM not in cache
  –receive compromise key list

Figure 4: Network Key Generation and Distribution Summary

2 .3 .1 .1 .2 Overview – Memory Partitioning

The Memory Manager mediates all memory transactions including those associated with the movement of all key material to/from the utilizing processes. The following overview sections are organized as a discussion of each of the five memory blocks shown managed by the Memory Manager.

As seen in the concept diagram of Figure 5, the CM key memory is divided into isolated red and black partitions. There is a black memory partition and four red memory partitions. The four red memory areas are (1) a partition where new locally generated symmetric key material is temporarily stored before being placed in cache, (2) a small red partition which holds the master public/private key pair of the subject CM, (3) a partition where keys that have been received from the network or DTD and decrypted are stored pending their negotiation and use per call set up protocol, and (4) the 64k fast access cache partition which facilitates the low latency key agile encryption/decryption of the data packets. A Memory Manager controls access to the memory partitions, and a Monitor/Alarm function insures that the keys used meet the requirements for strong keys and that no failure has occurred in the key supply process.

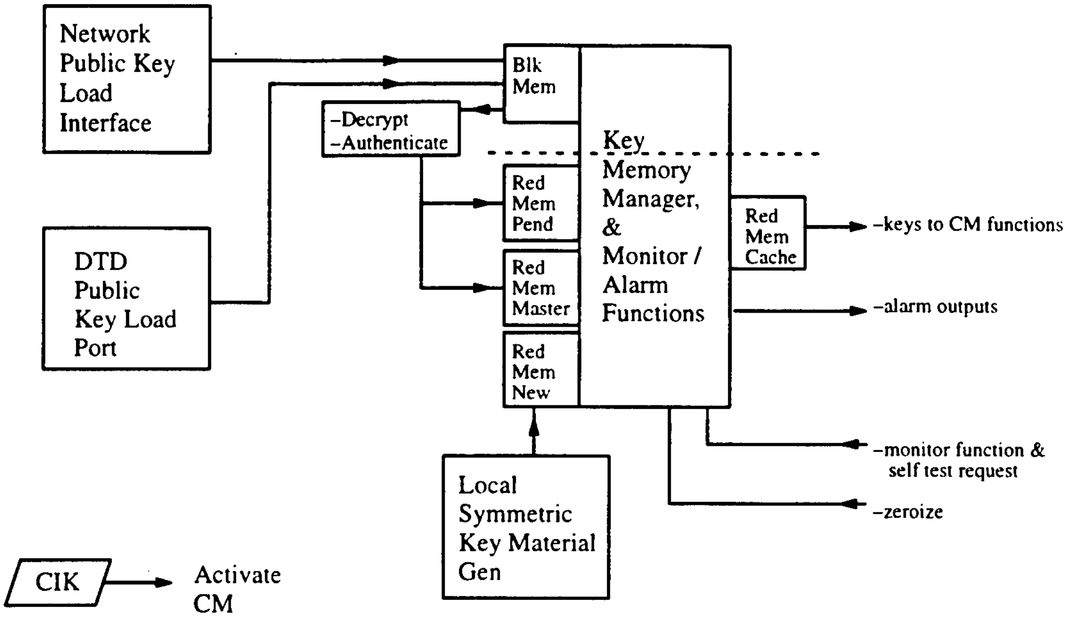


Figure 5: SHARE CM, Key Memory Management Concept

2 .3 .1 .1 .3 Overview – Wired 1st Master CM Private Key

In order to get the key load process started, an initial master key must be installed in the CM by a primitive process. The CM contains a factory wired private key (or prom encoded key) for this purpose. Initial accesses to the CM must be via the CM public key corresponding to the factory wired private key. In the event that the CM's public/private master key pair approach the end of their cryptoperiod, a subsequent key transfer may install a new master key pair using the current master key pair as the decryption/certification agent.

2 .3 .1 .1 .4 Overview – CIK

The CIK is a physical key inserted into the CM/SR to initiate a rigorous self test, and after a successful 'GO' result from the self test, to enable the CM's cryptographic functions and place the CM/SR on–line. The CIK contains an active element (IC chip) which, via public key cryptography, authenticates itself to the CM, including its permission to act as the ignition key.

2 .3 .1 .1 .5 Overview – Network Key Pairs via DTD Distribution

Each SHARE authorized CM/SR in the network must have a public/private key pair issued by the Key Distribution Center (KDC). The key pair is generated and supplied by the KDC and the associated EKMS infrastructure. One method that the CM can use to accept key material from EKMS is via a DTD key fill device. The keys transported by the DTD are in black form, having been encrypted with the CM's public key. They are loaded from the DTD into the CM black memory partition. Subsequently, the keys are decrypted with the CM private key and placed in red storage for use.

2 .3 .1 .1 .6 Overview – Network Key Pairs via Network Distribution

The network CM key pairs distributed by the DTD as described in Section 2 .3 .1 .1 .5 above, may also be distributed via the network itself. The CM key pairs so distributed by the network are similarly protected by the CM's public key cryptography and initially loaded into the black memory partition. Such key loads must be of finite size, and in this CM, key agile capability for communication with 64k other CMs is provided. If the network is larger than 64k SANs and a SAN host requests communication with a 'new' CM that has not been loaded into active memory, the CM requests the public key of this 'new' CM from the Certificate Authority (CA) as depicted in Figure 4. The reply by the CA contains the requested 'new CM' information which is protected and authenticated by the public key cryptography of the CM and CA.

In early SHARE system implementations, the traffic to/from the CA will be minimal as there are few secure SANs and their corresponding SR/CMs in place to require CA service. As the number of SHARE secure SANs becomes very large, there is a point where the amount of CA traffic increases and unacceptable latencies and even blockages can occur. Multiple CAs are then provided on the net to reduce a given CA's traffic to acceptable levels, with each CA serving a smaller group of SAN SR/CMs, possibly identified and constrained by organizational, geographical, or political boundaries. The additional administrative communication then required between the CAs is small compared to the traffic eliminated. In concept, an additional tier of CAs may be further added to accommodate the key material distribution needs of extremely large networks. In this way SHARE provides for a scalable network key distribution system as the nets continue to grow in size.

2 .3 .1 .1 .7 Overview – Session (Symmetric) Key Gen

Symmetric session key material is generated locally within the CM. The session keys are used as Transmission Encryption Keys (TEKs) for protection of the data transferred between CMs. The session symmetric keys are negotiated between the CMs during the Key Establishment protocol which is a part of the Secure PacketWay Call Set Up process. These negotiations are protected via the public key cryptography of the CMs. Note that if the session keys were supplied externally rather than generated at the time of need, and if there were 64k SANs to be accommodated in the network, then over 2 billion keys would be required to insure privacy between all communicating CM pairs. Additional considerations relevant to the local generation of session keys are outlined in Section 2 .3 .1 .2 .6 .

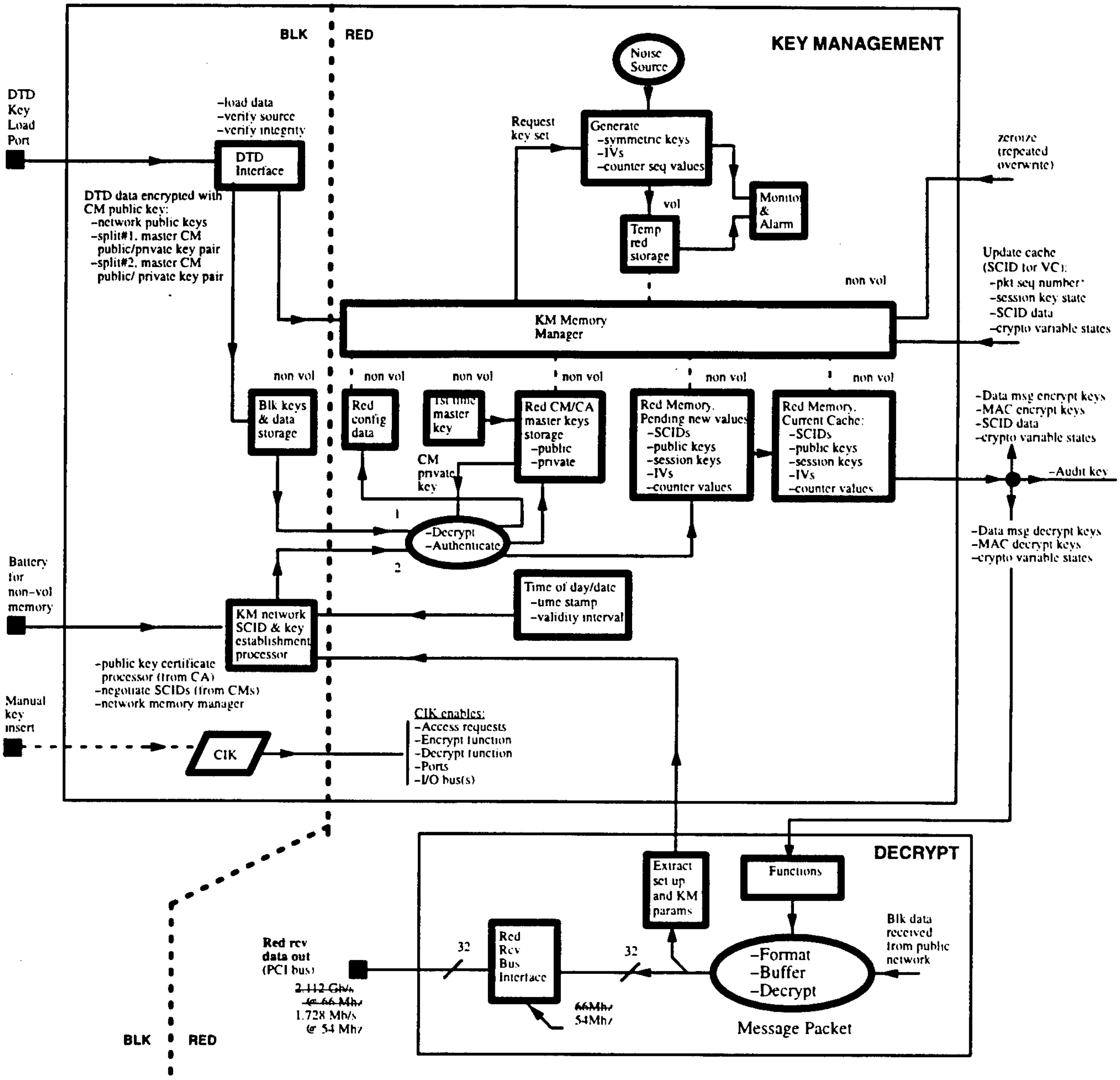


Figure 6: SHARE CM, Key Management Summary
(Logical Partitioning)

2 .3 .1 .2 CM Keying Scheme

Figure 6 expands the concept diagram of Figure 5 to show more key management detail. The figure should be referred to in the following subsections which describe various aspects of the keying scheme.

2 .3 .1 .2 .1 Configuration Data and Master Key Concepts

Two small memory partitions in Figure NO TAG are the 'Red Configuration Data' partition (not mentioned in the overview above), and the '1st Time Master Key' partition. The configuration data is the set of data defining the technical attributes of the CM. Such data includes software/hardware revision level. classification levels available (Type 1, ..., TS, ... etc), encryption and MAC algorithms available, cryptographic modes supported, key(s) lifetimes in effect, etc. Configuration data is the database that a CM uses to negotiate a SCID with a call recipient. For communication between CMs to occur, a compatible set of attributes must exist in each CM's database. Although the need to negotiate a compatible configuration with a recipient is trivial (and even unnecessary) if there are only a few CMs in the net, i.e. just store the configuration of all CMs, the negotiation process increases in value as the network is scaled to a large number of CMs and it becomes burdensome to attempt to store the configurations (and their updates). Negotiation of a compatible configuration is then the preferred call set up protocol.

The '1st Time Master Key' storage area holds the CM master key required to get the key management process started. This key is wired in (via prom) during production. The first loading of black key material uses this key to decrypt and convert the black keys to usable red keys. Subsequent key distribution may include a new master key which is used for the decryption of new key loads. The '1st master key' is actually a half of a split key, with the key load supplying the other half. Decrypting of the first key load requires that the supplied half key match the wired in half key. Subsequent master keys are also transported and stored in split form before their combination and decryption in the CM.

2 .3 .1 .2 .2 DTD Key Load Port

A DTD may be used to transport black key data from the Tier 3 Comsec Account to the CM/Secure Router. This black key data has been encrypted in the public key of the CM. The DTD port is seen to interface with the Black memory partition and the Memory Manager as described in the Overview above. The DTD interface is a serial port which accomplishes the loading of data, verification of the source validity, and verification of key data integrity. The key data furnished by the DTD includes the public keys and associated certificates for the CMs in the SHARE network. Up to 64k key sets may be transferred in this way. To accommodate nets of larger size, the Network Key Load Port is used as described below.

2 .3 .1 .2 .3 Network Key Load Port

If the usable network is to be scalable to an unlimited number of CMs, then an alternative to the storage (by each CM) of all network CM public and session keys, security parameters, and SCIDs must be put in effect. The solution here is to establish the keys at call set up. However, if the public, session, and MAC keys are supplied by a Certificate Authority (CA) for each call, the CA delays incurred each time a call is made or each time a session key expires is considered an operating burden. This is especially true if highly classified data requiring short session/MAC key lifetimes are a large part of the traffic.

The solution here is to only ask the CA for a new recipient's public key (when not in the cache database), to generate the symmetric session/MAC keys locally, and to transfer the session/MAC keys as the during a key establishment protocol which is part of the call set up. The 'easily generated' session/MAC keys are quickly available and need not consume large amounts of CM storage.

The call originator generates and supplies the session keys to the called recipient via the key establishment protocol, which is protected by public key cryptography. If the intended recipient's public key is not in the CM's 64k database, a Certificate Authority (CA) is contacted for the recipient's public key. The CA supplies the public key of the requested CM via a digitally signed certificate (the CM must have received the CA's public key by a trusted path), and the public key and identity of the recipient are thus validated, and communication can take place.

In this manner, new public keys are received (and 'old' ones discarded) over the network, and new session/MAC keys are locally generated and supplied (and 'old' ones discarded) at call set up.

2 .3 .1 .2 .4 The Red Pending Memory

When it is necessary to destroy and replace the public keys of one or many network CMs, the idea of a 'pending' memory is useful. By distributing the updated keys to a pending memory, and then executing a transfer of these keys to the active cache at a given time, delays in key establishment can be kept to a minimum. By this method it is not necessary to query the CA at call set up for the recipient CM's new public key. The key will have been obtained 'off line', and communication set up may proceed without incurring CA delays.

2 .3 .1 .2 .5 The 64k Cache

The red 64k cache is the memory partition containing all key and SCID material established and used in recent communications. It can be quickly accessed to perform key agile, low latency data communications with each associated CM. When the cache is full and traffic with a 'new' CM is requested, the keys and SCIDs are generated and negotiated as described above, and stored in this cache in place of that occupied by an 'old' CM.

2 .3 .1 .2 .6 Session Key Generation and Temporary Storage

As discussed in Section 1 .6 .1 , the number of session/MAC keys required can be very large even in modest nets. A net of 1000 CMs (SANs) requires approximately 500,000 session key sets in order to provide unique key sets to every possible pair of communicating CMs. And each CM has a different key load. If these keys are to be supplied by a centralized service, then their generation, distribution, destruction, and management is a time and resource consuming task. A better solution, as discussed in Section 1 .6 , is to generate the session key material locally. These keys are symmetric, and with the proper design care, may be generated in a CM rather easily.

Conceptually what is required for local CM generation is (1) a means for generating a random (or strong pseudorandom) number of 128 (max) bits, (2) a means for testing and selecting/deselecting strong/weak keys, and (3) a means for monitoring the key generation process for integrity and generating alarms and alarm responses when integrity falls below the established standard. These session key sets are generated via hardware or software implementations which take place in parallel with the on–line packet encryption/decryption processes. It is thus useful to have a temporary red storage partition in which to accumulate these self generated keys for their later (rapid) recall and use. This is referred to as the red 'New' memory storage partition.

2 .3 .1 .2 .7 Red / Black key Identification

The identification of all key material by its red / black classification is specifically supplied in the table in Section 2 .3 .2 . A graphical representation of the key material, although more general and not as complete, is provided in the 'Key Management' block of Figure 6.

2 .3 .2 Number Of Keys Required

✓ *2–address the key requirements (how many different keys are used by the item).*

✓ *3–identify who will supply the key to the user.*

2 .3 .2 .1 CM Keys Required, Storage Provided, and Supply Source

A key type within the CM goes through three stages before it is used in communications with another CM. The key staging design concept is discussed later in Section 2 .11 .1 , The Function of Each CM Key . The staging concept recognizes the flow of key through the CM. All keys used in the CM are listed in the following table. Columns in this table identify the key type, key length, memory partition where each key is stored, and the supply source. The flow of each individual key through the memory partitions can be seen by following along a horizontal line in the table. A 'key' for the abbreviations used in the table is located at the end of the table.

KEYS / CRYPTO VARIABLES vs MEMORY PARTITIONING:

		——Memory Partition——					
	Key space		————Red————				Govt/Int
Type of Key Material	Size	Blk	Mast	Pend	New	Cache	Supplied
(1) PUBLIC KEYS & CONFIGURATION DATA CATEGORY:							
DTD remote CMs' public keys/certs	64k x128B	b	.	p	.	c	G
DTD remote CMs' SCIDs	64k x128B	b	.	p	.	c	G
DTD CM configuration data	128 x32B	b	.	p	.	.	G / I
DTD CM public/private key	2 x128B	.	m	.	.	.	G
DTD Certificate Authority public key	1 x128B	b	.	p			G
DTD split#1 CM public/private keys	1 x128B	b	m	.	.	.	G
DTD split#2 CM public/private keys	1 x128B	b	m	.	.	.	G
Netw remote CMs' public keys/certs	64k x128B	b–	.	p–	.	.	–
Netw remote CMs' SCIDs	64k x128B	b–	.	p–	.	.	–
Netw CM configuration data	128 x32B	b–	.	p–	.	.	–
Netw CM public/private key	2 x128B	b–	m	.	.	.	–
Netw Certificate Authority public key	1 x128B	b–	.	p–			–
1st master private key	1 x128B	.	m	.	.	.	G
(2) *SPECIAL ACCESS* SESSION VALUES CATEGORY:							
session keys (last VC state)	16k x32B	.	.	p	.	c	I
session IVs	16k x32B	.	.	p	.	c	I
sequence numbers (last VC state)	16k x32B	.	.	p	.	c	I
header MAC keys	16k x32B	.	.	p	.	c	I
full message MAC keys	16k x32B	.	.	p	.	c	I
session SCIDs	16k x32B	.	.	p	.	c	N
(3) *TOP SECRET* SESSION VALUES CATEGORY:							
session keys (last VC state)	16k x32B	.	.	p	.	c	I
session IVs	16k x32B	.	.	p	.	c	I
sequence numbers (last VC state)	16k x32B	.	.	p	.	c	I
header MAC keys	16k x32B	.	.	p	.	c	I

full message MAC keys	16k x32B	.	.	p	.	c	I
session SCIDs	16k x32B	.	.	p	.	c	N
(4) *SECRET* SESSION VALUES CATEGORY:							
session keys (last VC state)	16k x32B	.	.	p	.	c	I
session IVs	16k x32B	.	.	p	.	c	I
sequence numbers (last VC state)	16k x32B	.	.	p	.	c	I
header MAC keys	16k x32B	.	.	p	.	c	I
full message MAC keys	16k x32B	.	.	p	.	c	I
session SCIDs	16k x32B	.	.	p	.	c	N
(5) *SBU* SESSION VALUES CATEGORY:							
session keys (last VC state)	16k x32B	.	.	p	.	c	I
session IVs	16k x32B	.	.	p	.	c	I
sequence numbers (last VC state)	16k x32B	.	.	p	.	c	I
header MAC keys	16k x32B	.	.	p	.	c	I
full message MAC keys	16k x32B	.	.	p	.	c	I
session SCIDs	16k x32B	.	.	p	.	c	N
(6) *OTHER* VALUES CATEGORY:							
DTD audit public key	1 x128B	b	.	p	.	c	G
DTD audit randomize vector	1 x32B	b	.	p	.	c	G
Netw audit public key	1 x128B	b–	.	p–	.	.	–
Netw audit randomize vector	1 x32B	b–	.	p–	.	.	–
CIK	2 x128B	b	.	p	.	.	G
temp red storage for self gen keys	64k x6 x32B	.	.	.	n	.	I

B –1 byte, 8 bits

b –key space required in 'Black' memory partition
m –key space required in 'Master' (red) memory partition
p –key space required in 'Pending' (red) memory partition
n –key space required in 'New' (red) memory partition
c –key space required in 'Cache' (red) memory partition

. –no memory space required
– –key supplier accounted for elsewhere in this table

G –Government supplied key material
I –Internally generated (by the CM) key material
N –Negotiated material (with the called recipient)

2 .3 .2 .2 Keys Required – Summary of CM Storage Required

A summary of the above memory partition requirements and implementation is provided below:

	x128B	x32B	Total
Black memory partition	128k + 4	0k + 131	~16MB
Master (red) memory partition	0k + 7		~ 0MB
Pending (red) memory partition	128k + 4	(64k x6) + 129	~32MB
New (red) memory partition		(64k x6)	~16MB
Cache(red) memory partition	128k + 1	(64k x6) + 1	~32MB
			~16MB Black
			~80MB Red

The key spaces in the above tables were based on the following base (single channel) requirements. It is seen that ample space has been reserved to allow for labels, error detection bits, and possible future extension of key lengths:

#bits	#bytes	Key Type
1024b	128B	–public key lengths
1024b	128B	–private key lengths
1024b	128B	–Security Context IDs (SCIDs)
256b	32B	–symmetric session and MAC key lengths
256b	32B	–IVs and counter sequence values
–	64k	–total number of virtual channels (key sets) accommodated by the key agile CM at a given time (65,536)

2 .3 .3 Key Lifetime Discussion

The table in the above Section 2 .3 .2 defines the key material required by a CM at a given point in time. Replacement key material (and destruction of 'old' key material) is required (1) within a time period that is dependent on the classification level of the data to be encrypted by the key and which is chosen to limit the amount of ciphertext available to an attacker, and/or (2) within a time interval selected to limit the quantity of traffic compromised if a key is uncovered, and/or (3) whenever the possibility of key compromise is detected. Items (1) and (2) are determined by user and network security policy, however, these time intervals should be orders of magnitude less than the period of the selected key. By determining the keys' (real time) period, an upper bound on the selected cryptoperiod can be defined.

In the SHARE system, session data may be transferred at rates up to 1.76 Gb/s. At this rate, an 80 bit session key has a cryptoperiod of 22e+6 yrs, and a 90 bit session key has a cryptoperiod of 23e+9 yrs, e.g. for a 90 bit key, [(2e+90)/(1.76e+9)]/[60x60x24x360]=23e+9 yrs. The most recent estimated age of the universe is 12e+9 yrs.

A more meaningful upper bound on useful key lifetime is based on a cracker's ability to uncover a key in a certain period of time. This requires the estimation of current computer computational power, the increase in computational power over the key lifetime, and the amount of resources that a determined and wealthy cracker may commandeer or develop. Skipjack is the secret government algorithm employed in the Capstone and Clipper chips. It has been published to have an 80 bit key and can be used in ECB, CBC, OFB, or CFB modes. A Skipjack review panel postulated several Skipjack cracking machines and their effectiveness (from Catlett, "Cryptography" course notes, March 1996):

1. "1993": –using an 8–processor Cray YMP performing 89K encryptions per second and costing $15M, would require 1 billion years.
2. "Future": –a future implementation of 100,000 paralleled RISC processors performing 100K encryptions each and costing $50M, would require 4 million years.
3. "Speculative": –a future implementation of 1.2 billion special purpose $1 chips, operating with a 1 GHz clock and performing 1 encryption per cycle, would require 1 year.

A key implementation of more than 80 bits of course requires a proportionate increase in the estimated cracking time and/or resources made available.

The 256 b (32B) session and MAC key lengths allowed for in the SHARE CM memory are thus seen to contain substantial reserve memory space even with respect to the requirements of an aggressive future security policy.

The conclusion arrived at from these estimates is that the lifetime of SHARE keys of 80 bits or greater need not be based on the key cryptoperiod or the related cracking abilities of committed adversaries, at least for the foreseeable future. Rather, key replacement intervals can be selected to limit the amount of plain text data divulged in the event of key compromise by any means. As always, the caveat must be added that these estimates are based on brute force cracking attempts and presume that there is "no breakthrough in the crypto analysis of the particular encryption algorithm." Again however, as far as the SHARE design is concerned, the key lengths provided for include substantial reserve length to which additional protection may be added in the event that a particular attack is discovered successful.

The following key replacement intervals are offered for the reader's consideration only, and do not represent a recommendation. A recommendation can be made only in consideration of the risk if the data is uncovered.

Key Lifetime	Key Type
1 message	–session and related keys for special ultra high priority data
1 week (TBD)	–session and related keys for Top Secret data
1 month (TBD)	–session and related keys for Secret data
1 year (TBD)	–session and related keys for SBU data

2 .3 .4 Supporting Components Required

✓ *6–identify other components required to enable the item to operate securely (fill device, CIK, Smart Key, etc).*

✓ *7–identify the above component's source (if its not part of the system proposed).*

There are five (5) devices that the CM is required to interoperate with. These five devices enable the CM to operate securely in the SHARE network environment:

	Type of Interface	Source	Function
1.	Direct	GPE*	–DTD (AN/CYZ–10V3) (the SHARE network may also perform the key fill function provided by this device)

2.	Direct	SHARE	–Secure Router
3.	Direct	SHARE	–Battery
4.	Direct	SHARE	–CIK
5.	Indirect	SHARE	–SHARE Network Manager –Network Management –Certificate Authority –Audit Manager –Email Manager –Trusted Network OS

*GPE Government Purchased Equipment

1. The DTD is a standard EKMS Tier 3 component. It is an automated key fill device which directly interfaces to the CM as described in this KMP. This device can be supplied/purchased from the government for use in authorized operations.
2. The Secure Router contains the CM which performs the cryptographic services and is described throughout this KMP.
3. The battery is a removeable component of the Secure Router and serves to provide power to functions which must be provided as non–volatile. The battery type has not yet been selected but for most applications is expected to be a sealed rechargeable type.
4. The CIK is the device used as the last step in 'turning on' the CM/Secure Router and placing it on–line. Although the table lists the 'source' as 'SHARE' as it is presumed to be supplied as a SHARE developed item, the CIK in fact will be virtually identical to that used in present COMSEC equipments, e.g. the STU–III Secure Telephone Unit.
5. The network services listed are to be provided by the SHARE environment and are thus considered an indirect interface in that they are logical rather than physical interfaces. These services have not yet been designed, although they will take the form of the corresponding services provided by EKMS and the MISSI upgrades.

2 .4 NET STRUCTURE

✓ *–define net structure in detail*
✓ *–depict net layout graphically and describe with supporting verbage.*
✓ *–if applicable, identify the point of key generation and paths of electronic key flow.*
✓ *–if appropriate, indicate where key is RED and where key is BLACK.*

Information on Net Structure is provided throughout this document. Section 2 .1 .3 describes "The System In Which the Item Will Be Used", i.e. the SHARE*HPSC Network. Figure 3 is a graphical depiction of "An Example SHARE Network Structure." Verbage is provided in that section which describes the net structure and layout. Further detail is included in Reference 2 , "SHARE*HPSC System Requirements." In this section, discussion focuses on the net structure elements which manage and control the generation and distribution of key material. As such, there is substantial discussion overlap with Section 2 .7 , Distribution.

As illustrated on the right hand side of Figure 7, CM keys (public/private) are supplied via the EKMS/MISSI infrastructure. This infrastructure provides for the ordering, distribution, and management of Comsec key material to approved users, e.g. DOD and civil agencies, government contractors, etc. The keys are ordered from and generated by the Central Facility at the Tier 0 level. Tiers 1 and 2 comprise the Comsec Material Control System. These tiers consist of the authorities that manage, control, document, and coordinate the transfer and activation of key material, and the privileges assigned to cryptonet members. Changes in the configuration of individual nets are also documented and reported to net members. Tier 3 represents the Comsec end user.

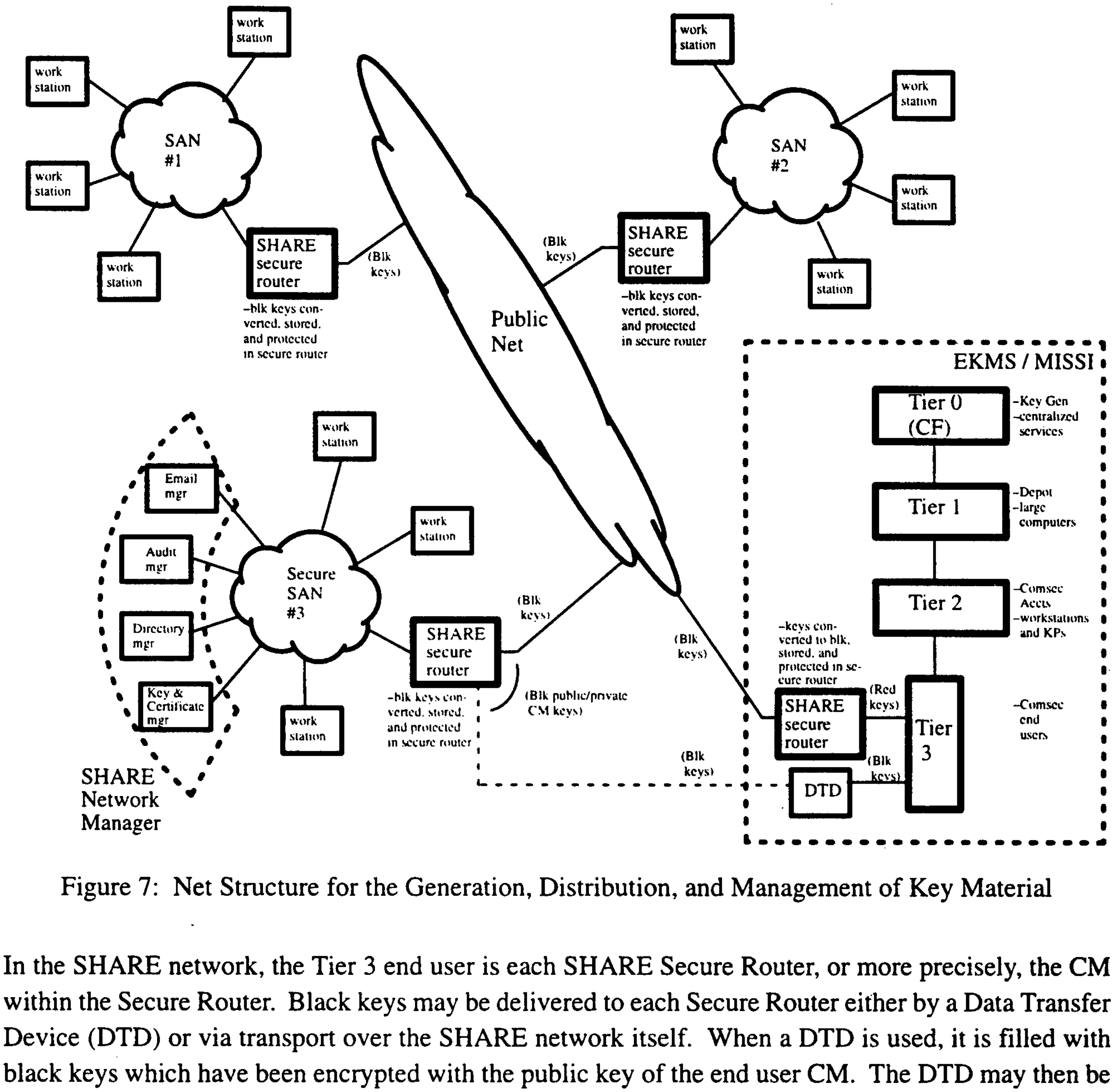


Figure 7: Net Structure for the Generation, Distribution, and Management of Key Material

In the SHARE network, the Tier 3 end user is each SHARE Secure Router, or more precisely, the CM within the Secure Router. Black keys may be delivered to each Secure Router either by a Data Transfer Device (DTD) or via transport over the SHARE network itself. When a DTD is used, it is filled with black keys which have been encrypted with the public key of the end user CM. The DTD may then be physically transported to each of the Secure Routers in the network, and via their DTD fill ports, the DTD loads the black keys into the CM. Instead of transporting the DTD to each of the Secure Routers, the DTD may only load the keys into one of the elements of the Network Manager, i.e. the Key and Certificate Manager. Each individual CM would then receive its key load via network distribution. This method requires longer call set up times when the CM first communicates with another CM, but it reduces the resources that need to be committed to key distribution.

The Net Structure, provides for three methods of key distribution:

1. A DTD, filled with black keys via the EKMS/MISSI infrastructure, is used to transport and load these keys into each of the Secure Routers in the SHARE network.
2. A DTD, filled with black keys via the EKMS/MISSI infrastructure, is used to transport and load these keys only into the 'Key and Certificate Manager'. The SHARE Network Manager then facilitates the distribution of these keys to the other Secure Routers.
3. Red keys generated and processed via the EKMS/MISSI infrastructure are sent to the SHARE Secure Router which interfaces the EKMS. The red keys are encrypted in the SR, and the resulting black keys are distributed to the 'Key and Certificate Manager' over the SHARE network. The SHARE Network Manager then facilitates the distribution of keys to the other Secure Routers.

In Method #1, red keys supplied by EKMS/MISSI are encrypted by the public key of the using Secure Router, and the DTD is filled with the resulting black keys for transport to the router. Since a DTD is used to fill all Secure Routers in the network, key packages encrypted with each of the Secure Router's public key's are required, and a separate key package must thus be transferred by a DTD to each router. This method is viable in the early stages of SHARE network deployment where there are relatively few such key packages required, and where their physical transport by DTD is operationally acceptable.

In Method #2, red keys supplied by the EKMS/MISSI infrastructure are encrypted by the public key of the 'Key and Certificate Manager', and the DTD is filled with the resulting black keys for transport to this network agent. Since the DTD is used to transport and load keys only into one agent, the demands placed on manual key transport and load is reduced to a minimum. However, the development and certification of a secure key exchange algorithm to be used for distributing these keys over the SHARE network requires development. This key distribution method must therefore follow method #1 in availability.

In Method #3, the DTD is eliminated, a SHARE Secure Router is interfaced to the EKMS/MISSI infrastructure, and keys are distributed to the "master Secure Router" over the SHARE network. Keys are further distributed to the other Secure Routers as done in Method #2. Method #3 thus requires the development of a SHARE Secure Router interface to the EKMS/MISSI infrastructure, and would follow Method #2 in availability.

All three key distribution methods thus require the transfer of only black key material. This black key material is stored in the each Secure Router's Black Key Storage area. The conversion from black to red key within the CM is described in another section of this KMP.

The conversion of key material from black to red by the Secure Router CMs implies the need for a shared key between the CM and the key supply device, i.e. either the DTD or the network supply agent. Since these keys are to be supplied by the EKMS/MISSI infrastructure, the first such key used to decrypt the first key exchange must be transported to the Secure Router by primitive/manual means. In this document, this key is referred to as the "first master key". Thereafter, key distribution may take place by one or more of the three methods listed above.

2 .5 ACCESS CONTROL

✓ *–define in detail the controls governing access to cryptographic keys within the item.*

✓ *–address how access will be authorized and validated to ________ cryptographic key.*

✓ *–request*

✓ *–generate*

✓ *–handle*

✓ *–distribute*

✓ *–store*

✓ *–use*

The three sub–paragraphs below address CM key access control, and how access to EKMS key material and locally generated key material will be "authorized and validated to request, generate, handle, distribute, store, and/or use cryptographic key."

2 .5 .1 Key Access Control Within the CM

The keys within the CM are generated, accessed, and routed to the appropriate CM functions as described in Section 2 .3 .1 Description of Keying Scheme , Section 2 .4 Net Structure , and Section 2 .7 Distribution. Access to the keys is granted by the CM only after the CM determines that the traffic request is consistent with security policy. This determination is based on the data sensitivity and other security labels that accompany the traffic request. The CM is seen to implement the Reference Monitor concept in that all such cryptographic service requests are mediated by the CM's associated security kernel. The requesting host supplies a 'source SCID' containing the security associations that are relevant to the data transfer, e.g. source data security level, source host security classification, recipient security classification, source and recipient SCID lifetimes, etc (see Section 2 .7 .4 for a more complete description of SCID content). The CM verifies the supplied data for accuracy and determines its consistency with respect to network security policy. If the originator and recipient SCIDs are compatible, the CM allows access to the needed key material and communication between the two CMs and the two hosts take place. If the CM determines that there is an incompatibility, the CM replies to the requesting host that the connection has been disallowed. The CM's monitoring and auditing functions are also activated to inform and record the failed traffic request.

2 .5 .2 Key Access Control of EKMS Supplied Material

The EKMS procedures that are in place will be followed for the supply of public/private keys for the network CMs. This includes procedures that enable the request, generation, handling, distribution, and storage of the key material. Very briefly, a COMSEC account will be established at Tier 2 for key distribution to the CM at Tier 3. The basic functions of each tier in the KM heirarchy are outlined in Section 1 .4 . The means of use of this CM public and private key material is provided in Section 2 .3 .1 .2 .

2 .5 .3 Key Access Control Of CM Generated Material

Session keys and related session key material are locally generated within the CM. The key material for a session with another CM is negotiated as part of the call set up protocol. The call originator generates and supplies the session key material after the recipient's certificate has been validated and the necessary access permissions verified. As described in Section 2 .3 .1 .2 .6 , the required key material is locally generated and supplied as required from a 'new' red storage memory partition which has been reserved for the storage of locally generated symmetric keys. Access to these keys has been described above in Section 2 .5 .1 , Key Access Control Within the CM .

2 .6 ACCOUNTING

✓ *–identify accounting data of interest.*

✓ *–address how the accounting data will be________ to document cryptographic key_______ within,*

✓ *–collected*
✓ *–maintained*
✓ *–updated*
✓ *–transferred*
✓ *–used*

✓ *–generation*
✓ *–distribution*
✓ *–storage*
✓ *–use*
✓ *–destruction*

or in association with, the item.

✓ *–description shall differentiate between automatic vs human actions required.*

The accounting data of interest is included in the Control Vector (CV) associated with each key and as described in Section 2 .10 .3 , Key Identification Within the CM . As described there, the CV is a 'long label' containing all of the security relevant information associated with a particular key.

The CV, i.e. the accounting data, is created at the time of key generation and/or activation as indicated within the list of parameters provided in the referenced paragraph. The CV is stored, distributed, and used concurrently with the key itself as described throughout this KMP. It is also destroyed when the key is destroyed.

The CV (accounting) data is processed as the key itself is processed through its various stages of use within the CM. For a discussion of the key stages concept, see Section 2 .11 .1 , The Function of Each CM Key, and Figure 9, Key Memory Staging . As the key is maintained, updated, transferred, and used, so is the CV (accounting) data. Since the session keys are created as needed and destroyed as required by their lifetime attribute, there is not a 'collection' of keys and key data that represents all keys used from the point time of the Secure Router going operational. Nor is there a means for extracting present or past keys or key data from the CM. The exception to this is when there is a need to store audit data. As seen in Figure 2 SHARE CM Signal and Functional Summary, audit interrupts occur and related information is stored upon a key change, keys valid change, and a keys zeroized condition.

The operations described are performed automatically under the control of the CM operating system. The exception is the monitor and audit functions which occur either automatically or manually by the operator at the Network/Audit Manager control console.

2 .7 DISTRIBUTION

✓ *–define in detail how cryptographic key will be distributed to and translated within the item.*

✓ *–indicate when, in the process, key is RED and when key is BLACK.*

✓ *–address how cryptographic key will be identified during this process.*

✓ *–describe how this distribution process will insure the integrity of the cryptographic key from point of origin to point of destination.*

There are two classes of key material that are distributed to/from the CM, i.e.
(1) the public/private keys for each of the network CMs (and their associated certificates, CVs, and SCIDs, and (2) the session and MAC keys (and their associated CVs and SCIDs).

The class (1) keys above are supplied via the EKMS infrastructure and distributed to each CM either by a DTD or via the SHARE network itself. Class (2) keys are generated locally by the CM itself, and transmitted to the recipient CM via public key cryptography during the call set–up protocol. Key distribution to the CMs has been described in Section 2 .3 Keying Scheme, Section 2 .4 Net Structure, and this Section 2 .7 Distribution. An important concept in the supply and establishment of the keys between the CMs is that of the SCID, or Security Context Identifier. Some detail on SCID data content is also provided below.

2 .7 .1 Distribution and Translation of CM Public/Private Key Pairs

The Class (1) CM public/private key pairs may be supplied via a DTD device as described in Section 2 .3 .1 .2 .2 , DTD Key Load Port, and/or via the Network Load Port described in Section 2 .3 .1 .2 .3 . The net structure which supplies the Class (1) keys from the EKMS infrastructure is discussed in Section 2 .4 , Net Structure . The keys supplied by either the DTD or network have been encrypted with the CM's public key, must be decrypted with the CM's private key, and are therefore considered to be black keys during distribution.

As depicted in Figure 7, keys are loaded into the DTD at the Tier 3 level and accounted for in an appropriate Comsec account. The keys are stored in the DTD in black form for manual transport to the CM. At the CM they are loaded into black (temporary) storage until the key load is complete. The keys are then decrypted with the CM private key and stored in the red 'pending' memory for use when requested. The black keys are considered installed when the the DTD is returned to the Comsec account and the key transfer verified. Activation of the keys, i.e. transfer of the keys from red 'pending' storage to red 'cache' then takes place throughout the network under control and at the command of the SHARE Network Manager. All CMs and the CA are thus coordinated to have the same active public/private key pair database. Note from the table in Section 2 .3 .2 that key storage (memory) partitioning allows for keys of different classification, e.g. special, TS, S, etc. to be activated at different times as determined appropriate by the network security policy.

The network key supply path to the CM is also depicted in Figure 7. A SHARE Secure Router connected to Tier 3 as shown supplies black keys to the end user CM and/or to the CA. These black keys are first stored in CM black memory, and then decrypted with the CM's (or CA's) private key and placed into 'red pending' memory as depicted in Figure NO TAG. Again, activation of the keys into red 'cache' memory is controlled by the SHARE Network Manager.

2 .7 .2 Distribution and Translation of Locally Generated Symmetric Keys

Session symmetric keys are generated within each CM. The call originating CM selects the session keys to be used from one of two available sets. The session keys are selected either (1) from the CM's red cache if existing there, i.e. if a previous communication with the same recipient has occurred and the necessary data retained, or (2) from the red 'new' key set if there is no cached key material for this recipient. Note that it cannot be guaranteed that a previously negotiated session key/SCID has not been overwritten by the intended recipient. The recipient CM may have had to reallocate that memory space to service its more recent traffic requirements. In the event that a CM 'erases and overwrites' a negotiated key/SCID that still has available life, a signalling message is sent to the partnered CM advising that succeeding communications must negotiate a new session key/SCID. In this way the red cache supports the high speed key agile functions of the CM without requiring 'unlimited' memory and/or without requiring the slower speed key/SCID negotiations to take place for every communication attempt. The keys transferred to the recipient are encrypted with the recipient CM's public key, must be decrypted with the recipient CM's private key, and are therefore considered to be black keys during distribution.

2 .7 .3 Key Identification and Integrity During Distribution

Each key contains a label which specifies the parameters associated with the key. The parameters and keys are encrypted together with the public key of the CM designated to receive the key load. In the case of public/private key pair distribution by the DTD, the keys are loaded into the DTD in black form, and loaded from the DTD into the CM in black form. In the case of public/private key pair distribution by the network, the keys and parameters are similarly in black form and are decrypted with the CM's private key before storage in the red 'pending' storage partition. In the case of symmetric session key distribution to the recipient (established at call set–up), the keys are protected by the public/private key pair of the destination CM.

2 .7 .4 SCID Content

During the call set–up period which precedes first time communication between two CMs, a compatible SCID is identified and negotiated between the communicating CMs. This SCID contains the security attributes, including keys, that the communicating entities will share in the data transfer. The source CM begins the SCID negotiation on receipt of a traffic request from a host node on its local SAN. The source CM's 'SCID offer' is based on the destination address and an expected set of security attributes in that CM. The following minimum security attributes are established in the negotiated SCID:

1 . –Session encryption algorithm and mode
    –Session key and IV
2 . –Header–MAC encryption algorithm and mode
    –Header–MAC key and IV
3 . –Full_message MAC encryption algorithm and mode
    –Full_message MAC key and IV
4 . Key and IV lifetimes (or time of key change)
5 . Lifetime of this SCID
6 . Replay protection (time stamp, accepted window size for keys, sequence numbers)
7 . Source address(es) of the SCID (may be a wild card if the SCID is shared)
8 . Sensitivity label (of protected data)
9 . Hosts' security classification
10 . Crypto synch field (if required)

2 .8 GENERATION

✓ *–define in detail (if applicable) the key generation process by which the item produces the cryptographic key.*
✓ *–specify the ________ used to generate the cryptographic key.*
✓ *–algorithm*
✓ *–equipment*
✓ *–system*

Symmetric session key material is generated locally by the call originator CM, supplied to the call recipient as part of the call set–up protocol, and used by the CM in the data packet and MAC encryption processes.

The generation of key material is done by classified algorithms approved for the generation of Type 1 and/or Type 2 keys. The keys will be generated by firmware and/or hardware on a separate independent module, or on the CM module itself. There are two basic types of schemes considered for the generation of this key material, i.e. a hardware and a firmware based scheme. Where speed of generation is important, and as shown in Figure NO TAG, a white noise source is used as the basis of the random number generator. The noise source bandwidth is consistent with the key length and the allotted key generation time period. In cases where there is time for computation, a software based complexity–theoretic generator is used to create the random numbers, i.e. RSA, BBS, etc. In either case, the random numbers (keys) are generated and stored in the red 'new' memory partition as a cache for their rapid access as required by the CM encryption and call set–up functions.

2 .9 RECOVERY

✓ *–describe in detail the recovery process by which secure comm can be restored in the event of the loss or compromise of cryptographic key.*

✓ *–description shall differentiate between auto, electrical vs physical distribution methods identified in the recovery process.*

2 .9 .1 Specific Key Compromise

When key compromise is discovered, or if a specific quantity of keys is lost due to any means, i.e. not a complete loss of the SHARE network key database, the Network Manager compiles a 'Compromised Key List' and distributes this list to all CMs. The compromised keys are erased (overwritten) in each CM. Each CM replies to the Network Manager that the 'Compromised Key List' was received and the necessary erasure procedure executed. If public cryptography keys are involved in the compromise, the CMs' public/private key pairs are also supplied, either by DTD or network distribution. If session keys are involved in the compromise, instead of supplying the CMs with a 'Compromised Key List', the Network Manager supplies the CMs with a "Compromised Communication Pairs List'. On receipt of this list, each CM purges its current key memory of the session key material used (and to be used) in communications between the compromised pair.

2 .9 .2 Complete Key Loss

Complete CM key loss may occur due to key zeroization or other unplanned event, e.g. non–vol battery failure. In this case it is necessary to restart the CM as if it were just leaving the factory – in fact the required procedure is to return the unit to the factory for audit analysis and key reinitialization. The recovery process is therefore the start–up process described in previous Sections 2 .3 Keying Scheme, 2 .4 Net Structure, and 2 .7 Distribution .

2 .9 .3 CM Alarm, Monitor, and Audit Functions

In the case of complete CM key loss due to zeroization via any of the detected alarms, CM monitor and audit data has been made available to the Network Manager to support the analysis of events leading up to the key loss. The alarms detect any event affecting, or potentially contributing to, encryption or key related compromise. Key zeroization occurs due to detected unauthorized access, tampering, contributors to conductive or radiated emissions, and other elements as appropriate for NSA certified systems and equipments and as described in related classified documents.

Listed within the Alarm, Monitor, and Audit blocks of Figure 2 are the functions processed by each of these blocks. Figure 8 illustrates the heirarchical organization of the Alarm/Monitor/Audit functions and of the response methodology. The Alarm function reacts instantaneously to detected intrusions. Data associated with the Alarm function is also sent to both the Monitor and Audit functions. The Monitor function stores Alarm and 'other' data which is considered indicative of possible imminent compromise. Data acquired by the Monitor function provides to a system operator near real time data permitting an assessment of the status of the CM. Monitor data may be collected as commanded by an operator, or automatically, via the Network Manager's operating system. The Monitor data is polled frequently by the Network Manager. The Audit function stores within the CM that data which can assist the CM/Secure Router's intrusion analysis. Note that all CM Audit data is also made available to the Network Manager via the Monitor function. The Audit function stores all relevant data of interest and is polled infrequently by the Network Manager or examined directly in the laboratory.

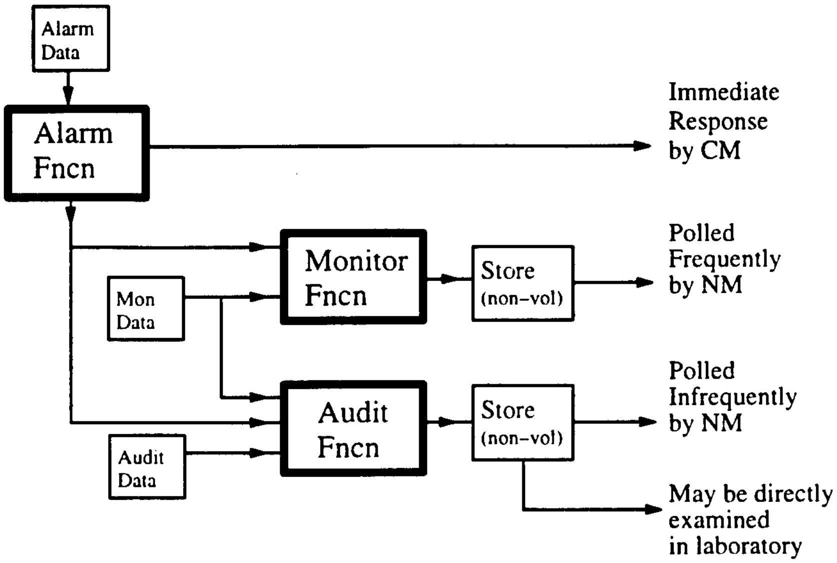


Figure 8: Alarm/Monitor/Audit Heirarchical Operation

The Alarm function is backed up by battery to permit its responses to be executed even in the event that power is removed. The Monitor function is also backed up by battery to enable the Network Manager to acquire such data even with power removal – note that this capability also requires that selected portions of the encrypt/transmit circuits must also be temporarily powered.

2 .10 STORAGE

- ✓ *–describe in detail how the item stores the cryptographic key from receipt (or physical load) to destruction.*
- ✓ *–explain how the storage process will ensure the integrity of the cryptographic key throughout this process.*
- ✓ *–define the storage capacity of the item for cryptographic key*
- ✓ *–define how the item will identify the cryptographic key during its storage life.*

The reader should refer to the diagrams and text associated with the Key Management block diagram of Figure 6 and the tables in Section 2 .3 .2 , Number of Keys Required , as an aid in understanding the discussion in the following sub–sections.

2 .10 .1 Key Storage, With Integrity – From CM Receipt to Destruction

Public/private key material loaded via the DTD port is placed in black temporary storage. These keys are decrypted with the private key of the CM and placed in red 'Pending' storage. The keys are then transferred to red active storage (cache) at a designated time consistent with similar transfers being performed by the other network CMs. The red cache is a high speed memory partition designed to support the low latency, key agility demands of the CM. The red memory storage areas are protected against compromise by equipment and access alarms which are described in Figure 2 SHARE CM Signal and Functional Summary, and Section 2 .9 .3 CM Alarm, Monitor, and Audit Functions . The destruction of a key set may be commanded by the Network Manager, or it may be initiated by CM internal process as a result of key lifetime expiration. When the entire key set is to be zeroized, the Memory Manager overwrites all key memory locations with random data.

2 .10 .2 CM Key Storage Capacity

The key storage requirements are detailed in the tables in Section 2 .3 .2 , Number of Keys Required. In summary, CM storage is required for ~16MB of black key material and ~80MB of red key material. In order to provide for clean memory segmentation/partitioning and ease of memory management, and to allow for possible features requiring memory to be added in the future, the CM will provide the capability to extend the black memory capacity by an additional 4MB (25%) and the red memory capacity by 16MB (20%).

2 .10 .3 Key Identification Within the CM

Keys stored in the CM are stored in conjunction with a 'Control Vector' (CV). The CV is a 'long label' containing all of the security relevant information associated with a particular key. The following lists the minimum set of parameters associated with a CV. This data provides operational information that provides for proper key usage (including access control), and provides some history for 'use analysis' by the Network Manager. It is also a listing of the accounting data as requested in Section 2 .6 .

Operational Parameters:

1 . Sensitivity level of the key (special, TS, S, etc)
2 . Type of key (symmetric, public, private)
3 . Key mode (red/black, new, active, pending, destruct)
4 . Lifetime of the key (time of key change, key lifetime)
5 . Key length (number of key bits, number of check bits)
6 . Key check bits algorithm (simple parity, CRC, etc)
7 . Key valid for use flag (strong key verified)
8 . Key usage flags (broadcast, single destination)

Historical Parameters:

9 . Method of generation / version number (internal H/W, internal S/W, external EKMS)
10 . Date / time of (key generation, first use)

2 .11 USAGE

✓ *–define in detail the function/use of each cryptographic key used by the item*

✓ *–explain how the item ensures the integrity of the cryptographic key during its use.*

2 .11 .1 The Function of Each CM Key

As extracted from the detailed listing of the table in Section 2 .3 .2 .1 , CM Keys Required, Storage Provided, and Supply Source , the keys used may be grouped into five categories corresponding to their memory partitions, i.e. one black and four red partitions:

1 . Black (memory Stage 1)
2 . Master (memory Stage 1)
3 . Pending (memory Stage 2)
4 . New (memory Stage 1)
5 . Cache (memory Stage 3)

A key type within the CM goes through three stages before it is used in communications with another CM. The key staging design concept is illustrated in Figure 9. The boxes in the figure represent the five memory partitions. The four inputs at the left of the figure represent CM key types based on CM access, i.e. via the DTD port, network distribution, factory wired, or locally generated.

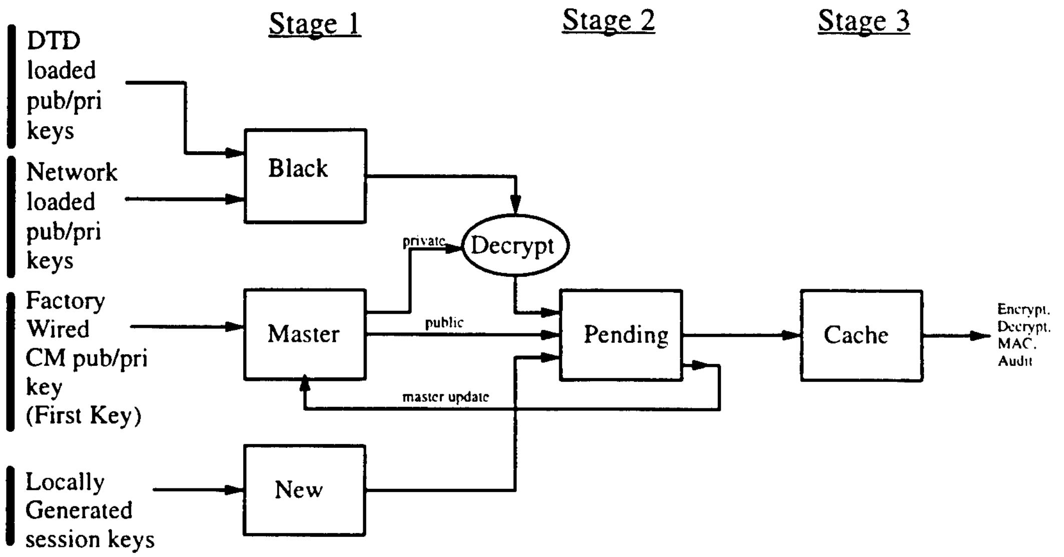


Figure 9: Key Memory Staging

The memory staging concept helps to associate and visualize the flow of key through the CM. All keys used in the CM are listed in the tables of Section 2 .3 .2 .

In this section, the function of each key and its associated security parameters are described in terms of its CM input classification, i.e. in terms of the four input types illustrated in Figure 9, and the key's ultimate use in the CM. Information stored in memory which is not key material related is not discussed, e.g. configuration data. In the descriptions below, the terminology corresponds to that in the tables of Section 2 .3 .2 .1 , and the (#)s preceding the paragraph correspond to the categories in those tables.

The purpose of the 'Black' keys is actually accomplished prior to their storage in the CM, i.e. to provide for their secure transport to the CM. The 'Black' storage partition is reserved for public/private keys generated/supplied by the EKMS infrastructure and loaded from the DTD port or via network distribution. These keys have been encrypted with the public key of the CM, and are decrypted by the CM using its private key before placing in red 'Pending' storage.

2 .11 .1 .1 The Function of the PUBLIC KEYS & CONFIGURATION DATA (1):

-DTD/Netw remote CMs' public keys/certs
-DTD/Netw remote CMs' SCIDs

The 'DTD/Netw' prefix for these keys indicates that they may be supplied via either the 'DTD port' or the 'network interface'. These are the public keys and certificates for the network CMs which the originating CM will communicate with. They also serve to identify and authenticate SHARE authorized CMs. The public keys are used to encrypt communications between CMs during the call set up negotiations. For example, during the call set up process, the calling CM sends the recipient CM session key material (described later) encrypted in the recipient's public key and signed with the caller's private key. The recipient confirms in a reply message (encrypted in the calling CM's public key and signed with the recipient's private key), that the offered session key material is acceptable. If it is not acceptable, negotiations continue until a common database is arrived at. Continued unacceptability may occur as the result of incompatible security attributes in their available SCID dictionary and the call attempt is terminated with appropriate signalling by the caller and recipient.

If additional keys corresponding to network CMs which are not included in the initial key load are required, they are obtained from the CA as described below.

–DTD/Netw CM public/private key

This is the public/private key pair of the subject CM. These are the subject's public cryptography keys that are used to establish communications and negotiate a SCID with a recipient CM.

–DTD/Netw split#1/2 CM public/private keys

As a double measure of protection, the distribution of the public and private key pair for the subject CM above is done in two steps with split keys.

–DTD/Netw Certificate Authority public key

When communications is required with the CA, the information is encrypted with the CA's public key.

–1st master key

A 'first key' must be installed in the CM by primitive means, i.e. the private key is wired at the factory and enables subsequent DTD or network key transport to be encrypted. After the key distribution process is established, the CM's public/private key pair can be updated as required either by DTD or network key load.

2 .11 .1 .2 The Function of the SESSION VALUES (2 – 5):

In this group there are four categories (2 – 5) , i.e. Special Access (2), Top Secret (3), Secret (4), and SBU (5). Each of the keys in these four categories is described below.

–Session keys (last VC (virtual channel) state)

Symmetric session keys are used to encrypt/decrypt the data packets transferred at high speed through the SHARE network. A given session key is used only for communication with the negotiated recipient CM and only for the designated key lifetime. When data is exchanged over a period of time between the CMs, and if a feedback encryption mode is used, e.g. CBC, CFB, OFB, or other counter modes, it is necessary to store the state of the session key of the virtual channel so that succeeding encryptions and decryptions will be in synch.

–Session IVs

If not protected, identical plaintext messages encrypt to the same ciphertext, or plaintext messages that begin the same will yield the same ciphertext up to the first difference. To prevent this, the first plaintext block of packet data is a block of random data, i.e. the Initialization Vector.

–Sequence numbers (last VC state)

Sequence numbers are placed in the transmitted data so that proper re–assembly may be accomplished by the recipient even though arriving packets may be out of sequence. Again, the last sequence number used in the VC communication is stored.

–Header MAC keys
–Full message MAC keys

This KMP allows the use of independent 'Header' and 'Full Message' MAC keys which are also independent of the session key. This allows the design of MAC functions for present and future CMs to take advantage of the 'currently best' design approach.

–Session SCIDs

Session SCIDs contain the parameters associated with a particular key and channel establishment. The SCID contents represent the common agreements by which communication between CMs takes place.

2 .11 .1 .3 The Function of the OTHER VALUES (6):

–DTD/Netw audit public key
–DTD/Netw audit initialization vector

The audit public key is used to encrypt audit data before its storage. The initialization vector insures the randomness of the initial and subsequent block(s) of ciphertext. Note that the network client that receives the audit data, i.e. the Network Audit Manager, is assigned the responsibility of decrypting the audit data. The Network Audit Manager thus has access to the 'audit private key' counterpart to the 'audit public key' used for encryption.

–CIK

The Crypto Ignition Key (CIK) is used in the last step of the process of turning on a CM/Secure Router and placing it in operation. Before CM activation with the CIK, power may be applied to the CM/Secure Router and non–cryptographic functions exercised as desired for off line test and verification. The insertion of the CIK and its authentication by the CM causes a complete self test and alarm test to be performed on the CM. A successful completion of this test allows the CM/Secure Router to be placed in operation. The CIK contains its own public/private key pair and certificate. This certificate serves to authenticate the CIK and confirms its special privilege to activate the full cryptographic capability of the CM.

2 .11 .2 Key Integrity Assurance by the CM During Key Use

The maintenance of key integrity as the keys are being used has been an implicit requirement throughout this KMP, and the methods employed to assure this integrity have been described where relevant. The following list summarizes some of the major factors contributing to the assurance of key integrity during its use by the CM:

System and Subsystem Design:

1. Key distribution is via Black keys.
2. Red / Black key isolation and memory partitioning.
3. CM Memory Manager mediates all access to cryptographic functions, i.e. Reference Monitor concept implemented by the associated security kernel.
4. Keys are not readable outside of the CM
5. CM cryptographic functions activated only with authenticated CIK and after successful self test and alarm test.
6. Keys erased in conjunction with network managed Compromised Key List

7. Secure Router physically located in System High SAN area
8. Tempest design discourages EMI attack (discussion not requested/included in this KMP)

Alarm, Monitor, Audit, Functions:

9. Alarm system zeroizes keys when possible crypto fault or compromise detected
10. Tamperproof features included in CM and Secure Router
11. Monitoring and auditing capability provides the data for analysis to strengthen the design

Key Design:

12. Key lengths discourage brute force attacks
13. IVs eliminate identical ciphertext
14. CVs identify key security associations and controls its use and accessibility
15. Key lifetime expirations automatically and/or manually executed

* Discussion not requested nor included in this KMP.

3 ACRONYMS AND ABBREVIATIONS

API	Application Program Interface
B	Byte (8 bits)
BBS	Blum, Blum, Shub (inventors, simplest/most efficient complexity–theoretic generator)
CA	Certificate Authority
CBC	Cipher Block Chaining
CF	Central Facility
CFB	Cipher Feedback
CIK	Crypto Ignition Key
CM	Cryptographic Module
CMCS	COMSEC Material Control System
COMSEC	Communications Security
COR	Central Office of Record
COTS	Commercial–Off–The–Shelf
CRC	Cyclic Redundancy Check
CSAC	COMSEC Account
CV	Control Vector
DES	Data Encryption Standard
DID	Data Item Description
DII	Defense Information Infrastructure
DMS	Defense Messaging System
DoD	Department of Defense
DTD	Data Transfer Device
ECB	Electronic Code Book
EKMS	Electronic Key Management System
FOUO	For Official Use Only (distribution limited to official use)
GPE	Government Purchased Equipment
GOTS	Government–Off–The–Shelf
H/W	Hardware
IDEA	International Data Encryption Algorithm
IETF	Internet Engineering Task Force
INE	In–line Network Encryptor
ITAR	International Traffic in Arms Regulations
IV	Initialization Vector
KDC	Key Distribution Center
KEK	Key Encryption Key

KG	Key Generator
KM	Key Management
KMP	Key Management Plan
KP	Key Processor
LAN	Local Area Network
LMCS	Lockheed Martin Communication Systems
LMD	Local Management Device
MAC	Message Authentication Code
MISSI	Multilevel Information System Security Initiative
MLS	Multilevel Secure (or Security)
NIA	Network Interface Adapter
NM	Network Manager
NOFORN	No Foreign Nationals (distribution not permitted to foreign nationals)
NOS	Network Operating System
NSA	National Security Agency
NTCB	Network Trusted Computing Base
OFB	Output Feedback
OS	Operating System
OSI	Open Systems Interconnection
PCI	Peripheral Component Interconnect
RSA	Rivest, Shamir, Adleman (inventors, the most popular public–key algorithm)
S	Secret
S/W	Software
SAN	System Area Network
SBU	Sensitive But Unclassified
SCID	Security Context Identifier
SHARE•HPSC	Secure Heterogeneous Application Runtime Environment for High Performance Scalable Computing
SR	Secure Router
TBD	To Be Determined (or Defined)
TCB	Trusted Computing Base
TCSEC	Trusted Computer Security Evaluation Criteria
TEK	Transmission Encryption Key
TGS	Ticket Granting Server
TS	Top Secret
VC	Virtual Channel

4 DEFINITION OF TERMS

SHARE•HPSC: Secure Heterogeneous Application Runtime Environment for High Performance Scalable Computing

Security: In the SHARE environment, cryptographic methods and open standards are used to ensure information survivability in the transmission of (multi–level) secure packets of information over a public network. The network security objectives are data integrity, confidentiality and authentication, and availability.

Heterogeneity: SHARE is the hardware and software fabric that provides interoperation between inherently dissimilar network components, i.e. between SANs of various types and security levels, and between host nodes having differing bandwidth and operating system requirements.

Application Runtime Environment: The provided high speed, low latency network environment enables applications to run across cooperating heterogeneous SANs in near real time. Application programs use traditional Application Program Interface (API) software to access the underlying network services.

High Performance: High performance here means high bandwidth, low latency, operation in an error environment, on–line and off–line testability, and controlled fault recovery.

Scalable: Scalability is the ability to increase or decrease the size and heirarchy of a connected network of SANs using fundamental SHARE•HPSC building blocks. Very large scale extensions to the SHARE network of SANs are possible using basic building blocks which include, but are not limited to, secure and non secure routers, local SANs, and high speed public network switches.

. . . . . .

Application Program Interface – API: Software that can be referenced by an application program to access underlying network services, i.e. a set of software commands that an application may make to a software module providing a defined service, e.g. a cryptographic service.

Audit: A function performed by a computer system (or network) which monitors and records selected system activities so that actions affecting security can be traced to the responsible party. Audit information must also be protected. Examples of events that are included in network audits are the establishing and dropping of a network connection, the occurrence of lost, misrouted, or timed out data, valid and invalid log in attempts, and the failure of a network component.

Authenticate: To establish the validity of a claimed identity, e.g. data, sender.

Certificate Authority: A trusted central repository of public key information. The information needed to enforce the security policies of the cryptographic system may be provided by this authority or, in more rigorous implementations, in other servers dedicated to access control.

Certification: The technical evaluation of a system's security features, made as part of and in support of the approval/accreditation process that establishes the extent to which a particular system's design and implementation meet a set of specified security requirements.

Commercial–Off–The–Shelf – COTS: Specifies standard commercially available hardware or software that can be purchased 'immediately' and without modification.

Communications Security – COMSEC: Protective measures taken to deny unauthorized information access derived through communications, and to employ a security policy to authenticate communication. (NCSC–WA–001–85).

Context: In an MLS system, a security context is the particular set of security definitions associated with the message, e.g. "a session key is assigned to each message based on its security context."

Degree of Protection: Computer data can be protected to varying degrees based on a multilevel security model and multiple subject authorization keys. SHARE provides for 65000 subject authorizations.

Electronic Key Management System – EKMS: EKMS is the government's COMSEC material logistics and provisioning system. The EKMS is cooperatively specified and developed by various government agencies, and is a unified network of systems and equipments independently developed by the member agencies. The unified EKMS provides COMSEC material managers with the means to plan, coordinate, control, and execute COMSEC material supply requirements through the shared use of the interoperable member systems.

Host: Any computer based system connected to the network and containing the necessary protocol interpreter software to initiate network access and carry out information exchange across the communications network. Example: A terminal is not a host because it does not contain the protocol software needed to perform information exchange; a workstation (by definition) is a host because it does have such capability.

Information Survivability: (See MISSI).

International Data Encryption Algorithm – IDEA: A very strong (at this time) symmetric key algorithm using a 128 bit key. IDEA is a part of PGP (Pretty Good Privacy), a freeware email security program.

International Traffic in Arms Regulations – ITAR: Regulations covering the export of arms, including cryptographic methods.

Internet Engineering Task Force – IETF: a subgroup of the Internet Activities Board. IETF supports the advancement of technologies through the cooperative work of representatives of the industrial forces in the computing industry. Information is disseminated, documents are published, and standards developed in the many working groups which are intended to give some direction to Internet development activities.

Key Management – KM: The generation, storage, distribution, deletion, archiving, and application of keys in accordance with a security policy. Key management includes ensuring that key values generated have the necessary properties, making keys known in advance to the parties that will use them, and ensuring that keys are protected as necessary against disclosure and/or substitution. All keys have limited lifetimes.

Key Management Plan – KMP: The plan is a description of the key scheme proposed for a crypto system and/or equipment. It describes the management of all key distributed through and employed in the system from the time it leaves the point of generation until it is destroyed. The plan is used to assist the government in the equipment's certification and in the determination of its compatibility and supportability by existing systems.

Local Area Network – LAN: A network, typically in the Mbps range, wherein all segments of the transmission are situated in an office, building, or campus environment. Ownership is by the user organization.

Lockheed Martin Communication Systems – LMCS: A SHARE•HPSC subcontractor.

Mandatory Access Control: A means of restricting access to objects based on the sensitivity (as represented by a label) of the information contained in the objects and the formal authorization (clearance) of the subjects to access information of such sensitivity.

Multilevel Information System Security Initiative – MISSI: MISSI is NSA's program for near term data security development. Products developed under this program are targeted for use by agencies of the DoD. ARPA's similar program is called Information Survivability and targets defensive information warfare needs not only for DoD, but for U.S. society as a whole. ARPA's program addresses the long–term data security needs and is not expected to offer near term solutions. (From Military and Aerospace Electronics, Aug 1995).

Multilevel Secure – MLS: A class of system that provides a capability for various levels and categories or compartments of data to be stored and processed in an AIS, and permits selective access to such material concurrently by users who have differing security clearances and need–to–know. The identification, segregation, and control of users and sensitive material on the basis of security clearance, material classification category, and need–to–know are essentially under automated control. Also see TCSEC.

Network Architecture: The conceptual description of the way communication is accomplished between data processing equipments at different sites. It also specifies the processors and terminals, protocols, and software that must be used.

Network Interface Adapter – NIA: A general term for any network interface adapter which is placed between a node and a network cable.

Network Trusted Computing Base – NTCB: The totality of protection mechanisms within a network, the combination of which is responsible for enforcing the network's security policy. The hardware, firmware, and software composing a network system that is responsible for enforcing a security policy.

Node: The point from/to which directed data originates/culminates, i.e. whatever can send and receive packets.

Object: A passive entity that contains or receives information, e.g. records, blocks, pages, segments, files, directories, directory trees and programs, as well as bits, bytes, words, fields, processors, video displays, keyboards, clocks, printers, network nodes, etc.

Open Systems Interconnection – OSI: A framework for network architectures consisting of 7 layers, e.g. physical, data link, network, transport, session, presentation, and application. The bottom three layers support the components of the network necessary to transmit a message, the next three layers generally pertain to the characteristics of the communicating end systems, and the top layer supports the end users.

"Orange Book": Document DOD 5200.28–STD, "Trusted Computer System Evaluation Criteria". Also see TCSEC.

PacketWay: PacketWay, previously known as MessageWay, is the network layer protocol used in SHARE•HPSC. Secure PacketWay is an extension that contains the security provisions utilized by SHARE•HPSC.

Peripheral Component Interconnect – PCI: An advanced high speed data bus, i.e. defined as 32/64 bit, 33/66 MHz, 132 to 528 MB/s.

Port: A number which is generally identified with a process running on a host, i.e. which application program is to receive the incoming traffic (allows multiple user programs to communicate concurrently with the one application program). A port is usually a small uP with its own separate clock, memory, registers, and often, a CPU (a full fledged micro computer).

"Red Book": Document NCSC–TG–005, Version 1, "Trusted Network Interpretation". Also see TCSEC.

Router: A device that employs the bottom three OSI layers to interconnect remote and/or dissimilar networks. The router segments network traffic based on the routing algorithm, the destination network layer address, a higher layer protocol, and/or the associated LAN facility in use.

Sanders: A Lockheed Martin Company. The prime SHARE•HPSC contractor.

Secure Router: A router which interconnects trusted hosts on their own subnetwork with external untrusted systems. The secure router provides security services, e.g. cryptographic functions, for the trusted hosts when they communicate via external untrusted systems.

Security Policy: The set of laws, rules, and practices that constrain and control security relevant activities in the management, protection, and distribution of sensitive information.

Subject: An active entity, generally in the form of a person, process, or device, that (1) causes information to flow among objects or, (2) changes the system state.

Subject Security Level: A subject's security level is equal to the security level of the objects to which it has both read and write access. A subject's security level must be dominated by the clearance of the user the subject is associated with.

System Area Network – SAN: SANs are packet networks made of point to point links with flow control and using source routes. Modern SANs have high throughput, low latency, and low error rate, and are usually confined within a room, building, or plant. A SAN is generally an 'independent island' incapable of direct intercommunications with other SANs and high performance LANs.

Trusted: The belief that a system meets its specifications, and in data security, pertaining to hardware and software systems that have been designed and verified to avoid compromising, corrupting, or denying sensitive information.

Trusted Computer Security Evaluation Criteria – TCSEC: A pair of documents developed by NCSC, i.e. DoD 5200.28–STD, the "Orange" book, and NCSC–TG–005, the "Red" book, setting a policy standard of a basic set of requirements which defines (evaluates) degrees of assurance in an AIS. The TCB and NTCB of a SHARE system are currently defined to be Class B2, "Mandatory–Structured Protection". B2 systems require discretionary and mandatory control for all subjects and objects, covert channels are addressed, the TCB must be carefully structured into protection critical and non critical elements, the TCB interface must be well defined and enabled to be subjected to thorough testing and review, strong authentication mechanisms are provided, trusted facility management is provided, and stringent configuration management controls are imposed. The "Red" book, "Trusted Network Interpretation", provides interpretations of the "Orange" book for trusted computer/communications network systems. The "Red" book thus extends the TCSEC "Orange" book to networks of computers and describes a number of additional security services that arise in conjunction with networks.

Trusted Computer System: A system that employs sufficient hardware and software integrity measures to allow its use for processing simultaneously a range of sensitive or classified information.

Trusted Computing Base – TCB: The totality of protection mechanisms within a computer system, including hardware, firmware, and software, the combination of which is responsible for enforcing the system's security policy.

Trusted Path: A mechanism by which a trusted person at a terminal can communicate directly with the TCB. This mechanism can only be activated by the person or the TCB and cannot be imitated by untrusted software.

Trusted Subnetwork: A subnetwork containing hosts and routers that trust each other not to engage in active or passive attacks and trust that the underlying communications channel, e.g. Myrinet, isn't being attacked.

Type 1: A classified cryptographic algorithm or device used in defense applications and approved by NSA for protecting classified information.

Type 2: A classified cryptographic algorithm or device that is approved by NSA for protecting sensitive unclassified information in systems involving intelligence, national security, and certain military activities.

Type 3: A Type 3 cryptographic algorithm or device is one that is approved as a Federal Information Processing Standard (FIPS) and is used for protecting sensitive unclassified information.

Type 4: Type 4 is a commercial cryptographic algorithm or device that is not based on Federal Information Processing Standards (FIPS). NIST will maintain a Computer Security Objects Register (CSOR) listing these items.

5 REFERENCES

SHARE DOCUMENTS:

1 . "SHARE•HPSC Network Security Architecture, Rev–1", Sanders, a Lockheed Martin Company, 13 June 1996, Document # PUBS–96–C30–W.

2 . "SHARE•HPSC System Requirements, Rev–1", Lockheed Martin, 18 Sep 1996, Document # PUBS–96–C37–W.

3 . "SHARE•HPSC Cryptographic Module Requirements, Rev–", (in preparation).

TECHNICAL DOCUMENTS FROM VARIOUS SOURCES:

4 . "Principles of Key Management", Fumy & Landrock, IEEE Journal on Selected Areas in Communications, Vol 11, No. 5, June 1993.

5 . "Guide to Understanding Trusted Facility Management", NCSC–TG–015, Library No. S–231,429, National Computer Security Center.

6 . "Automated Distribution of Cryptographic Keys Using the Financial Institution Key Management Standard", Balenson, IEEE Communications Magazine, September 1995

7 . "ANSI X9.17, American National Standard, Financial Institution Key Management (Wholesale)", American National Standards Institute.

8 . "ANSI X9.9, American National Standard for Financial Institution Message Authentication", American National Standards Institute.

9 . "DoD Information Security: Near Term Dissonance, Long Term Promise", Adams, Military and Aerospace, August 1995.

10 . "XOR MACs: New Methods for Message Authentication Using Finite Pseudorandom Functions", Bellare, Guerin, and Rogaway, Advances in Cryptology – Crypto 95 Proceedings, Lecture Notes in Computer Science Vol. 963, D.Coppersmith ed., Springer–Verlag, 1995.

11 . "Design of a Key Agile Cryptographic System for OC–12c Rate ATM", Stevenson, Hillery, Byrd, Gong, and Winkelstein, MCNC Internet Society Symposium on Network and Distributed System Security, IEEE Computer Society Press, 1995.

12 . "Technical Report Evaluation Criteria for Cryptography", (Temporary Draft), Common Criteria for Information Technology Security Evaluation, V 0.99b, 3/30/96.

13 . "DoD Trusted Computer Evaluation Criteria", CSC–STD–001–83, 1983, DoD 5200.28–STD, Library No. S225,711, National Computer Security Center, 1985, "Orange Book".

14 . "Trusted Network Interpretation, of the Trusted Computer Evaluation Criteria", NCSC–TG–005, 31 July 1987, "Red Book".

BOOKS:

15 . "Applied Cryptography", Schneier, John Wiley & Sons, Inc, second edition, 1996.

16 . "Computer Communications Security", Ford, Prentice Hall PTR, 1994.

17 . "Computer Security Basics", Russell and Gangemi, O'Reilly & Associates Inc, 1991

18 . "Information Security", Longley, Shain, Caelli, M Stockton Press, 1992

6 APPENDIX

6 .1 APPENDIX A. KMP PREPARATION INSTRUCTIONS:

The following "(KMP) Preparation Instructions" has been extracted from DI–OT–0021 "Key Management Plan", a DID which defines the requirements of a typical KMP required under government contract. These "Preparation Instructions" have been used as a template for the formal KMP presented in Section 2.

10 PREPARATION INSTRUCTIONS

10.1 General – The Key Management Plan will document in detail how key generated in support of the cryptosystem and/or equipment will be managed from the time it leaves the point of generation until it is destroyed.

10.2 Content Requirements – The Key Management Plan shall contain the following:

10.2.1 Introductory Information – Introductory information shall state the purpose of the item and shall identify the item as Type 1 or 2. It shall describe the system in which the item will be used. Terminology throughout the Key Management Plan shall be consistent with the national INFOSEC glossary.

10.2.2 Communications Architecture – The communications architecture shall specify the communications requirements (voice, data), describe the communication scenario (LAN, broadcast, point to point, telephone, etc.) and specify the minimum and maximum net sizes.

10.2.3 Keying Scheme – The keying scheme proposed shall be defined in detail. It shall: address the key requirements (how many different keys are used by the item); identify who will supply the key to the user; specify the distribution medium (physical, electronic, other); identify when key is RED and when key is BLACK; identify other components required to enable the item to operate securely (e.g. fill device, CIK, Smart Key, etc.) and the component's source (if it is not part of the system proposed).

10.2.4 Net Structure – The net structure shall be defined in detail. The net layout shall be depicted graphically and shall be described with supporting verbage. If applicable, the point of key generation shall be identified as well as the paths of electronic key flow. If appropriate, indicate where key is RED and where key is BLACK.

10.2.5 Access Control – The controls governing access to cryptographic key within the item shall be defined in detail. Address how access will be authorized and validated to request, generate, handle, distribute, store, and/or use cryptographic key.

10.2.6 Accounting – This section shall identify accounting data of interest and shall address how the accounting data will be collected, maintained, updated, transferred, and used to document cryptographic key generation, distribution, storage, use, and/or destruction within, or in association with, the item. The description shall differentiate between automatic vice (sic, i.e. vs) human actions required.

10.2.7 Distribution – The distribution information shall define in detail how cryptographic key will be distributed to and translated within the item. Indicate when, in the process, key is RED and when key is BLACK. Address how cryptographic key will be identified during this process. Describe how this distribution process will ensure the integrity of the cryptographic key from the point of origin to point of destination.

10.2.8 Generation – If applicable, define in detail the key generation process by which the item produces cryptographic key. Specify the algorithm, equipment, and/or system used to generate the cryptographic key.

10.2.9 Recovery – This section shall describe in detail the recovery process by which secure communications can be restored in the event of the loss or compromise of cryptographic key. The description shall differentiate between automatic, electrical vice (sic, i.e. vs) physical distribution methods identified in the recovery process.

10.2.10 Storage – This section shall describe in detail how the item stores cryptographic key from receipt (or physical load) to destruction. Explain how the storage process will ensure the integrity of the cryptographic key throughout this process. Define the storage capacity of the item for cryptographic key and how the item will identify the cryptographic key during its storage life.

10.2.11 Usage – The function/use of each cryptographic key used by the item shall be defined in detail. Explain how the item ensures the integrity of the cryptographic key during its use.

Document End:

SHARE*HPSC CRYPTOGRAPHIC MODULE, KEY MANAGEMENT PLAN

≈U.S. GOVERNMENT PRINTING OFFICE: 1998-610-130-81045

DISTRIBUTION LIST

addresses	number of copies
RALPH KOHLER AFRL/IFTC 26 ELECTRONIC PARKWAY ROME NY 13441-4514	2
SANDERS, A LOCKEED MARTIN COMPANY SIG PROCESSING CENTER, PTP02-B002 ADVANCED ENGINEERING & TECH DIV PO BOX 868 NASHUA NH 03061-0868	1
AFRL/IFOIL TECHNICAL LIBRARY 26 ELECTRONIC PKY ROME NY 13441-4514	1
ATTENTION: DTIC-OCC DEFENSE TECHNICAL INFO CENTER 8725 JOHN J. KINGMAN ROAD, STE 0944 FT. BELVOIR, VA 22060-6218	2
DEFENSE ADVANCED RESEARCH PROJECTS AGENCY 3701 NORTH FAIRFAX DRIVE ARLINGTON VA 22203-1714	1
RELIABILITY ANALYSIS CENTER 201 MILL ST. ROME NY 13440-8200	1
ATTN: GWEN NGUYEN GIDEP P.O. BOX 8000 CORONA CA 91718-8000	1
AFIT ACADEMIC LIBRARY/LDEE 2950 P STREET AREA B, BLDG 642 WRIGHT-PATTERSON AFB OH 45433-7765	1

ATTN: TECHNICAL DOCUMENTS CENTER 1
OL AL HSC/HRG
2698 G STREET
WRIGHT-PATTERSON AFB OH 45433-7604

US ARMY SSDC 1
P.O. BOX 1500
ATTN: CSSD-IM-PA
HUNTSVILLE AL 35807-3801

NAVAL AIR WARFARE CENTER 1
WEAPONS DIVISION
CODE 48L000D
1 ADMINISTRATION CIRCLE
CHINA LAKE CA 93555-6100

SPACE & NAVAL WARFARE SYSTEMS CMD 2
ATTN: PMW163-1 (R. SKIANO)RM 1044A
53560 HULL ST.
SAN DIEGO, CA 92152-5002

COMMANDER, SPACE & NAVAL WARFARE 1
SYSTEMS COMMAND (CODE 32)
2451 CRYSTAL DRIVE
ARLINGTON VA 22245-5200

CDR, US ARMY MISSILE COMMAND 2
REDSTONE SCIENTIFIC INFORMATION CTR
ATTN: AMSMI-RD-CS-R, DOCS
REDSTONE ARSENAL AL 35898-5241

ADVISORY GROUP ON ELECTRON DEVICES 1
SUITE 500
1745 JEFFERSON DAVIS HIGHWAY
ARLINGTON VA 22202

REPORT COLLECTION, CIC-14 1
MS P364
LOS ALAMOS NATIONAL LABORATORY
LOS ALAMOS NM 87545

AEDC LIBRARY 1
TECHNICAL REPORTS FILE
100 KINDEL DRIVE, SUITE C211
ARNOLD AFB TN 37389-3211

COMMANDER 1
USAISC
ASHC-IMD-L, BLDG 61801
FT HUACHUCA AZ 85613-5000

US DEPT OF TRANSPORTATION LIBRARY 1
FB10A, M-457, RM 930
800 INDEPENDENCE AVE, SW
WASH DC 22591

AWS TECHNICAL LIBRARY 1
859 BUCHANAN STREET, RM. 427
SCOTT AFB IL 62225-5118

AFIWC/MSY 1
102 HALL BLVD, STE 315
SAN ANTONIO TX 78243-7016

SOFTWARE ENGINEERING INSTITUTE 1
CARNEGIE MELLON UNIVERSITY
4500 FIFTH AVENUE
PITTSBURGH PA 15213

NSA/CSS 1
K1
FT MEADE MD 20755-6000

ATTN: OM CHAUHAN 1
DCMC WICHITA
271 WEST THIRD STREET NORTH
SUITE 6000
WICHITA KS 67202-1212

AFRL/VSOS-TL (LIBRARY) 1
5 WRIGHT STREET
HANSCOM AFB MA 01731-3004

ATTN: EILEEN LADUKE/D460 1
MITRE CORPORATION
202 BURLINGTON RD
BEDFORD MA 01730

OUSD(P)/DTSA/DUTD 2
ATTN: PATRICK G. SULLIVAN, JR.
400 ARMY NAVY DRIVE
SUITE 300
ARLINGTON VA 22202

MISSION
OF
AFRL/INFORMATION DIRECTORATE (IF)

The advancement and application of information systems science and technology for aerospace command and control and its transition to air, space, and ground systems to meet customer needs in the areas of Global Awareness, Dynamic Planning and Execution, and Global Information Exchange is the focus of this AFRL organization. The directorate's areas of investigation include a broad spectrum of information and fusion, communication, collaborative environment and modeling and simulation, defensive information warfare, and intelligent information systems technologies.